UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

CONSULTORIA EM SISTEMAS DE INFORMAÇÃO

FLEXIBLE WORKSPACE

OPERATING SYSTEMS PROVISIONING

Nuno Filipe da Luz Marques

Trabalho orientado pelo Prof. Doutor Thibault Nicolas Langlois
e coorientado por Ricardo Manuel Pires Karim Ahmad

TRABALHO DE PROJETO

MESTRADO EM INFORMÁTICA

2015

Agradecimentos

Começo por agradecer à Unisys (Portugal) Sistemas de Informação, pela oportunidade, e a todos os seus colaboradores, internos e externos, pela simpatia e cordialidade que sempre demonstraram.

Agradeço particularmente ao *Manager* António Pedro Cunha, ao Pedro Araújo, à Inês Ribeiro e aos elementos da equipa em que fui recebido, passados e presentes, nomeadamente, José Coelho, Jorge Bravo, Eduardo Santos, Eduardo Mendes e, mais recentemente, Paulo Lopes e Marcelo Costa.

Não poderia deixar de realçar e agradecer, em particular, o apoio do meu orientador, Prof. Doutor Thibault Nicolas Langlois e do coorientador na Unisys, Ricardo Ahmad, pelas dicas preciosas e pelas revisões e críticas ao meu trabalho.

Um agradecimento muito especial à minha mulher, Vera Silva, por todo o apoio, paciência e compreensão, pois o que era para ser apenas um Minor de um ano, acabou por progredir para uma Licenciatura e depois para um Mestrado.

Por último e não menos importante, a todos os familiares e amigos que contribuíram, direta ou indiretamente, para a conquista de mais esta etapa.

Se não puder voar, corra.

Se não puder correr, ande.

Se não puder andar, rasteje,

Mas continue em frente de qualquer jeito.

Martin Luther King.

Resumo

Este relatório apresenta o projeto realizado na empresa Unisys (Portugal) Sistemas de Informação S.A., no intervalo de 25 de Setembro de 2014 a 24 de Junho de 2015, com o objetivo pessoal de concluir o Mestrado em Informática na Faculdade de Ciências da Universidade de Lisboa.

Este projeto teve como finalidade a otimização de parte do processo de aprovisionamento de sistemas operativos, mais concretamente do processo de captura dos *Drivers*, que são posteriormente distribuídos para todas as máquinas cliente físicas, utilizadas pelos colaboradores da Organização CLIENTE.

Este processo era conduzido de forma maioritariamente manual, sendo por esse motivo muito moroso e consequentemente dispendioso, pois obrigava a deslocações internacionais frequentes, visto que implicava a presença física de um especialista, para acesso físico às máquinas de referência.

As máquinas de referência são modelos iguais aos usados pelos colaboradores, ligadas ao ambiente de produção, nas instalações do CLIENTE, para efeito de testes pré-distribuição e captura dos *Drivers* para o repositório.

É deste repositório que os *Drivers* são depois distribuídos para as máquinas cliente, durante o processo de aprovisionamento do Sistema Operativo.

Tendo em conta estes fatores, foi investigada e desenvolvida uma forma possível de otimizar este processo, combinando as ferramentas e tecnologias com as quais os restantes elementos da equipa já estavam familiarizados.

O principal objetivo foi reduzir, tanto quanto possível, a intervenção manual e a necessidade de presença física nas instalações do CLIENTE, permitindo ainda que, no futuro, qualquer um dos elementos da equipa consiga alterar, ou complementar, qualquer parte da solução desenvolvida, se necessário.

Palavras-chave: Espaço de trabalho flexível, Aprovisionamento de Sistemas Operativos, Qualificação de *Drivers*, Gestor de configurações, *Powershell.*

Abstract

This report presents the Project developed in Unisys (Portugal), between the 25 of September 2014 and 24 of June 2015, with the personal goal of concluding the Master of Science in Computer Science from the University of Lisbon, Faculty of Sciences.

This project's goal was the optimization of part of the Operating Systems provisioning process, specifically the Drivers capture process. These Drivers are Distributed to all physical client machines, used by Organization's collaborators.

This process was manually driven mainly, being for that reason very slow and consequently expensive, because it implied frequent international travelling, since it needed physical presence of a specialist for access to the Reference Machines.

The Reference Machines are Models like the ones used by the Organization's collaborators, that must be connected to the Production Environment, in CLIENT's facilities, for pre-distribution testing and Drivers capture to the repository.

It's from this repository that the Drivers are Distributed to the client machines, during the Operating Systems provisioning process at a later time.

Considering this factors, a possible optimization of this process was investigated and developed, combining the tools and technologies with which the remaining team elements where already familiar with.

The main goal was reducing, as much as possible, manual intervention and the need of physical presence in CLIENT's facilities, allowing also that, in the future, any of the team elements can change, or complement, any part of the solution, if necessary.

Keywords: *Flexible Workspace, Operating Systems Provisioning, Drivers Qualification, Configuration Manager, Powershell.*

Conteúdo

LISTA DE TERMOS TÉCNICOS XVIII

LISTA DE FIGURAS XXX

LISTA DE TABELAS XXXII

CAPÍTULO 1 INTRODUÇÃO 1

1.1 MOTIVAÇÃO 1

1.2 OBJETIVOS 1

1.3 CONTRIBUIÇÕES 2

1.4 ESTRUTURA DO DOCUMENTO 2

CAPÍTULO 2 PROJETO 5

2.1 CONTEXTO 5

2.1.1 Apresentação do CLIENTE e do Projeto implementado pela Unisys 5

2.1.2 Requisitos do Projeto implementado pela Unisys 9

2.1.3 Migração para a solução implementada 9

2.1.4 Solução implementada pela Unisys 12

2.1.5 Master Factory 16

2.1.5.1 Funções da *Master Factory* 16

2.2 OBJETIVOS DO TRABALHO, NO CONTEXTO DO PROJETO APRESENTADO E DAS FUNÇÕES DA *MASTER FACTORY* 21

2.3 PLANEAMENTO 22

2.3.1 Planeamento inicial do trabalho 22

2.3.2 Análise dos desvios ao planeamento 24

CAPÍTULO 3 TRABALHO REALIZADO 25

3.1 FERRAMENTAS USADAS 25

3.1.1 Microsoft Deployment Toolkit (MDT) 25

3.1.2 System Center Configuration Manager (SCCM) 2012 26

3.1.2.1 Hierarquia do SCCM 29

3.1.3 Windows Management Instrumentation (WMI) 30

3.1.3.1 Arquitetura do WMI 31

3.1.4 Powershell 36

3.1.4.1 *Cmdlets Powershell* específicos para o SCCM 41

3.1.4.1.1 New-CMDriverPackage 41

3.1.4.1.2 Get-CMDriverPackage ... 42
3.1.4.1.3 Start-CMContentDistribution ... 42
3.1.4.1.4 Get-CMCategory ... 43
3.1.4.1.5 New-CMCategory ... 44
3.1.4.1.6 Import-CMDriver ... 45
3.1.5 Command shell ... *46*
3.1.6 Console Registry Tool for Windows (REG.EXE) ... *47*
3.1.7 Windows PreInstallation Environment (Windows PE) ... *48*
3.1.8 Deployment Image Servicing and Management (DISM) ... *49*
3.1.9 Ferramentas de gestão de Drivers dos respetivos fabricantes ... *50*
3.1.9.1 HP *SoftPaq Download Manager* (SDM) ... 50
3.1.9.2 Lenovo *Update Retriever* ... 50
3.1.9.3 HP *System Software Manager* (SSM) ... 51
3.1.9.4 Lenovo *Thin Installer* ... 51
3.1.10 Double Driver ... *53*
3.1.11 PnPUtil ... *53*
3.1.12 Microsoft Signtool ... *54*
3.1.13 Microsoft Driver Verifier ... *55*
3.1.14 Microsoft Windows Debugger ... *57*
3.2 TRABALHO REALIZADO ... 58
3.2.1 Análise do problema ... *58*
3.2.2 Investigação e desenvolvimento da solução otimizada ... *62*
3.2.3 Implementação da solução ... *68*
3.2.3.1 Configuração genérica importante na *Task Sequence* ... 69
3.2.3.2 Tarefas desenvolvidas e implementadas ... 70
3.2.3.2.1 Validar condições de instalação da máquina cliente ... 70
3.2.3.2.2 Formatar disco ... 70
3.2.3.2.3 Instalar Sistema Operativo ... 71
3.2.3.2.4 Desativar atualizações automáticas do Windows ... 71
3.2.3.2.5 Instalar Drivers de rede ... 72
3.2.3.2.6 Instalar a Framework .NET ... 73
3.2.3.2.7 Instalar o Software do fabricante ... 74
3.2.3.2.8 Configurar o Software do fabricante ... 74
3.2.3.2.9 Instalar o Double Driver ... 75
3.2.3.2.10 Correr Software do fabricante ... 76
3.2.3.2.11 Obter informação do Hardware ... 77
3.2.3.2.12 Verificar informação do estado dos dispositivos de Hardware ... 78

3.2.3.2.13 Verificar assinatura digital dos Drivers ... 81
3.2.3.2.14 Testar Drivers (testes de stress com o Microsoft Verifier) ... 82
3.2.3.2.15 Correr Double Driver para backup dos Drivers para o repositório ... 83
3.2.3.2.16 Analisar DUMP (Microsoft Windows Debugger) ... 83
3.2.3.2.17 Importar Drivers para a Task Sequence ... 83
3.2.4 Testes, resultados e avaliação da solução ... *85*
3.2.4.1 *Testes efetuados* ... 85
3.2.4.2 *Testes não efetuados* ... 86
3.2.4.3 *Resultados obtidos* ... 86
3.2.4.4 *Avaliação* ... 87
CAPÍTULO 4 CONCLUSÃO ... 89
4.1 TRABALHO DESENVOLVIDO ... 89
4.2 DIFICULDADES ENCONTRADAS ... 89
4.3 CONCLUSÕES ... 90
4.4 TRABALHO FUTURO ... 90
ANEXO 1 ... 93
ANEXO 2 ... 99
ANEXO 3 ... 101
ANEXO 4 ... 103
ABREVIATURAS ... 105
BIBLIOGRAFIA ... 107

Lista de Termos Técnicos

***Active Directory*:** Tecnologia de serviços de diretório da *Microsoft*, que oferece forte integração com os sistemas *Windows* e extensibilidade através do *Identity Federation* (AD FS)**.** No projeto NGWP, o *Active Directory* é usado para dar cobertura aos seguintes serviços: Serviços de *Hosting* e *Network Operating System* (NOS), Serviços de Autenticação, Serviços de Autorização, Serviços de segurança (certificados, registo automático, etc), pesquisas de atributos, não existindo atualmente uma floresta global do *Active Directory* que dê cobertura a todo o seu perímetro.

***Active Directory Federation Services* (ADFS):** Componente de *Software* desenvolvido pela *Microsoft* que pode ser instalado em Sistemas Operativos *Windows Server* para disponibilizar acesso *Single Sign-On* aos sistemas e aplicações, aos utilizadores localizados fora dos limites da rede da organização. Usa um modelo de autenticação baseada em declarações para manter a segurança das aplicações e implementar identidade federada. Autenticação baseada em declarações é o processo de autenticar um utilizador baseado num conjunto de declarações acerca da sua identidade, contidas num *token* confiável. O *token* será emitido e assinado por uma entidade habilitada para autenticar o utilizador por outros meios e que é da confiança da que faz a autenticação.

***Application Catalog*:** Portal que dá acesso às aplicações disponíveis para instalação a cada utilizador**.**

AppLocker*:** Contém novas potencialidades e extensões que permitem criar regras para permitir ou bloquear a execução de aplicações, baseadas em identificadores únicos dos ficheiros, e especificar que utilizadores ou grupos podem correr essas aplicações, melhorando as caraterísticas e funcionalidade das Políticas de restrição de *Software.**

***Archetype*:** Em português: arquétipo, é algo que é considerado um exemplo típico ou perfeito de um tipo particular de pessoa ou coisa, por ter todas as suas características mais importantes. No contexto do projeto representa um modelo de utilização, em

termos tipo, Arquitetura e Sistema Operativo. Por exemplo, o Sistema Operativo é diferente para cada *archetype*:

- *Fat Client – Windows 7 Enterprise*;
- *Thin Client - Windows Thin* PC;
- HVD - *Windows Embedded based* ou *Linux*.

***Backbone*:** Nome atribuído à infraestrutura (componentes de *Hardware* e *Software*) central necessária para a entrega dos serviços do NGWP. No contexto de redes de computadores, o *Backbone* (traduzindo para português, espinha dorsal, embora no contexto de redes, *Backbone* signifique rede de transporte) designa o esquema de ligações centrais de um sistema mais amplo, tipicamente de elevado desempenho.

***BitLocker*:** A Encriptação de Unidade *BitLocker* é uma nova funcionalidade de segurança integral do *Windows* que fornece uma proteção considerável para o sistema operativo no computador e para os dados armazenados no volume do sistema operativo. O *BitLocker* garante que os dados armazenados num computador que esteja a executar o *Windows* permanecem encriptados mesmo que o computador seja adulterado quando o sistema operativo não está em execução. Isto ajuda a proteger contra "ataques *offline*", ataques efetuados desativando ou contornando o sistema operativo instalado ou removendo fisicamente o disco rígido para atacar os dados separadamente.

***Blue Screen*:** Ecrã apresentado nos sistemas operativos *Windows* em caso de erro grave de sistema. Poderão ocorrer se um problema grave provocar o encerramento ou reinício inesperado do *Windows*. Estes podem ser provocados por problemas de *Hardware* ou *Software*, podendo ser difícil resolvê-los. Por exemplo, quando o núcleo ou um *Driver* de dispositivo em execução em modo núcleo encontra um erro que não pode recuperar. A única ação segura que o sistema pode executar é reiniciar o computador, gerando possíveis perdas de dados, caso o utilizador não tenha guardado os seus documentos.

***Boot image*:** É um tipo de imagem de disco (consiste num arquivo único contendo toda a estrutura e conteúdo de uma unidade de armazenamento digital de dados, seja um HDD, CD, DVD, SSD ou outra. Uma imagem de disco geralmente é criada a partir da cópia setor-a-setor da media de origem, ignorando o sistema de ficheiros, e, dessa forma, replicando perfeitamente a estrutura e o conteúdo da unidade de armazenamento). Quando transferida para um dispositivo permite ao *Hardware* associado arrancar (“*boot*”).

***BranchCache*:** Foi criada para reduzir a utilização da ligação WAN e melhorar a resposta das aplicações dos funcionários de filiais que acedem ao conteúdo a partir de servidores em locais remotos. Para isso, os computadores clientes das filiais usam uma *cache* de dados, que pode ser distribuída localmente em computadores clientes (modo de *cache* distribuída) ou hospedada num servidor da filial (modo de *cache* hospedada).

Captura: Processo que consiste na recolha de uma coleção de ficheiros e pastas que replica a estrutura original de um computador existente, incluindo a estrutura de ficheiros e pastas do sistema operativo, criando um ficheiro ".wim" que é uma réplica do disco rígido onde estes se encontram.

***Central Administration Site* (CAS):** Em português, *Site* de Administração Central, deve ser instalado primeiro. Gere múltiplos Sites Primários e não tem clientes. É usado apenas para administração e *reporting*.

***Citrix XenApp*:** Sistema de entrega e gestão de aplicações *Windows* (SBC/D e SBC/A), da *Citrix*, como um serviço aos utilizadores em qualquer lugar e usando qualquer dispositivo. Esta infraestrutura de entrega de aplicações combina o que há de melhor em virtualização e *streaming*, aceleração de aplicações *Web*, segurança e visibilidade. O *XenApp* permite que seja gerida centralmente uma única instância de cada aplicação e entregue aos utilizadores para uso *online* e *offline*, oferecendo uma experiência melhor do que se estiver instalada.

***Cluster*:** É o nome dado a um sistema que relaciona dois ou mais computadores para que estes trabalhem em conjunto, no intuito de processar uma tarefa**.**

***Common Information Model* (CIM):** Modelo usado para descrever como representar objetos reais**.**

***Compliance*:** Conjunto de disciplinas para fazer cumprir as normas legais e regulamentares, as políticas e as diretrizes estabelecidas para o negócio e para as atividades da organização, bem como evitar, detetar e tratar qualquer desvio ou inconformidade que possa ocorrer**.**

***Configuration Item* (CI):** É um registo na CMDB, que consiste em informação acerca de cada *Configuration Item* e é mantido ao longo do seu ciclo de vida. Tipicamente

incluem *Hardware*, *Software*, edifícios/geografia, utilizadores e documentação formal tal como documentação de processos e *service level agreements*.

Configuration Management Database **(CMDB):** Representa a configuração autorizada dos componentes de um ambiente de TI de uma organização, que ajuda a entender a relação entre estes componentes e acompanhar a sua configuração. Guarda informação relativa a todos os componentes do Sistema de Informação. Contém os detalhes sobre os *Configuration Items* (CI) na infraestrutura, atributos importantes e relações entre CIs.

Deployment**:** Consiste no conjunto de processos necessários para entregar *Software* a um dispositivo, coleção de dispositivos, rede, ou a toda uma infraestrutura**.**

Distributed Management Task Force **(DMTF):** É uma organização que lidera o desenvolvimento, a adoção e a unificação de iniciativas e padrões de administração de área de trabalho e de *Internet* para ambientes empresariais,

Distribution Point **(DP):** Adequado para instalar em redes de banda baixa, suporta até 500 clientes. Funciona como centro de distribuição de conteúdos e permite aos utilizadores obter e executar *Software*, como aplicações, *Software Packages*, *Software Updates*, imagens de Sistemas Operativos e *boot images*. Podemos controlar a distribuição destes conteúdos através de opções de agendamento e limitação de largura de banda (“*bandwidth throttling*”).

Drivers**:** São uma parte importante do Sistema, pois atuam como intermediários entre este e os dispositivos de *Hardware*. Interpretam os sinais e facilitam a comunicação entre ambos. Sem os *Drivers* indicados, os dispositivos não funcionam corretamente ou não funcionam de todo.

Fat client**:** Cliente que, em modelos cliente/servidor, executa a maioria do processamento**.** Neste tipo de cliente não é necessária uma ligação continua ao servidor. Tal como no caso do *Thin client*, o termo refere-se usualmente a *Software*, mas mais uma vez é também usado para descrever computadores em rede. Uma das maiores vantagens dos *Fat clients* assenta no facto de alguns sistemas operativos e *Software* não serem capazes de correr em *Thin clients*, enquanto que nestes são, pois têm os seus próprios recursos.

Ficheiros ".cab": Arquivos ("*Cabinet Files*") do *Windows*, que guardam informação comprimida usada para a instalação de *Software*. Suporta compressão sem perda

("*lossless data compression*") e integra certificados digitais, que são usados para manter a integridade do arquivo.

Gestão de Incidentes: Processo ITIL que tem como principal objetivo restaurar a operação normal do serviço o mais rapidamente possível, minimizando os prejuízos à operação do negócio e garantindo assim o melhor nível de serviço e disponibilidade. A operação normal do serviço é definida dentro do acordo de nível de serviço, que é outro processo ITIL. A gestão de problemas identifica e remove causas de incidentes do ambiente de TI, através da sua análise, a fim de garantir a estabilidade máxima dos serviços de TI.

***Governance*:** Conjunto de processos, costumes, políticas, leis e regulamentos que regulam a maneira como uma empresa é dirigida, administrada ou controlada**.**

***Group Policy Object*s (GPO):** As configurações de Políticas são guardadas em *Group Policy Object*s. Há dois tipos de *Group Policy Object*s: local e *nonlocal*. Os *Group Policy Object*s locais são guardados nos computadores individuais. Apenas um *Group Policy Object* local existe num computador, e contém um subconjunto das configurações que estão disponíveis num *Group Policy Object nonlocal*. Configurações de *Group Policy Object* local podem ser sobrescritas pelas configurações *nonlocal* se estiverem em conflito; caso contrário, ambas se aplicam. *Group Policy Object*s *nonlocal*, os quais são armazenados num *Domain Controller*, estão disponíveis apenas num ambiente com *Active Directory*. Estes aplicam-se a utilizadores e computadores no *Site*, *Domain*, ou *Organizational Unit* (OU) com os quais o *Group Policy Object* está associado.

***Hypervisor* ou monitor de máquinas virtuais, em inglês: "*Virtual Machine Monitor*" (VMM)**: *Software*, *firmware* ou *Hardware* que cria e corre máquinas virtuais. É uma camada de *Software* entre o *Hardware* e o sistema operativo, que permite executar múltiplos sistemas operativos num mesmo equipamento, simulando cada sistema hóspede como se houvesse um *Hardware* dedicado ao mesmo.

Imagem: Coleção de ficheiros e pastas que replica a estrutura original de um computador existente, incluindo a estrutura de ficheiros e pastas do sistema operativo, ou seja, é uma réplica de um disco rígido. O *Operating System Deployment* (OSD) suporta o formato *Windows Image* (WIM), que é o formato das imagens capturadas na fase de captura. Um ".wim" é, portanto, uma coleção compactada de ficheiros e pastas.

***Information Technology Infrastructure Library* (ITIL):** É a abordagem à Gestão de Serviços em Tecnologias de Informação mais utilizada em todo o mundo. Baseia-se na recolha das melhores práticas existentes e fornece um enquadramento adaptável e flexível para gerir os Serviços de TI. Permite a criação de uma linguagem e conceitos partilhados entre os elementos das equipas técnicas da organização e com os seus utilizadores, fornecedores, subcontratados e parceiros, facilitando o entendimento e a definição de abordagens comuns. A utilização das recomendações do ITIL pode permitir a obtenção de reduções substanciais de custos, através da otimização da utilização das capacidades e conhecimentos de pessoas, de processos adequados e da tecnologia disponível. A gestão proactiva e a contínua melhoria de serviços preconizada no ITIL contribuem para o aumento da qualidade em simultâneo com a contenção dos custos globais.

***Light Fat client*:** É um *Fat client* mais limitado em termos de recursos**.**

***Managed Object Format* (MOF):** Linguagem usada para descrever classes *Common Information Model* (CIM). A forma recomendada de implementar novas classes WMI é em ficheiros MOF que são compilados para o repositório WMI usando o Mofcomp.exe.

Máquina Cliente ou dispositivo cliente ("*Client, Client Device, Client Computer, Workstation*"): Dispositivos usados pelos utilizadores finais, para as suas necessidades de trabalho. Correm tipicamente um Sistema Operativo *Desktop*.

Máquina de referência ("*Reference Machine*" ou "*Reference Computer*"): Computador que é configurado exatamente como queremos que os computadores dos utilizadores fiquem configurados. Uma imagem deste *reference computer* é capturada e gerado o respetivo ficheiro ".wim", sendo este importado para o *Deployment share*. Esta imagem é finalmente distribuída para os computadores dos utilizadores ou computadores alvo ("*target computers*").

***Master Images*:** Conjunto mínimo de componentes de *Software* (Sistema Operativo, *Software standard* ou do negócio, *Drivers*, agentes e portal) que tem de ser instalado numa estação de trabalho física ou virtual para permitir ao utilizador final aceder ao seu espaço de trabalho**.**

***Microsoft Application Virtualization* (App-V):** Plataforma integrada de virtualização que transforma aplicações em serviços virtuais geridos centralmente, que nunca precisam ser instalados e não causam conflito com outras aplicações, permitindo dessa forma utilizar as mesmas em qualquer lugar e através de qualquer rede e não apenas na rede corporativa**.**

Migração: Consolidação, atualização ou movimentação de todos os utilizadores de uma unidade de *Software* ou *Hardware* específicos, para outra versão, plataforma ou ambiente**.**

***Network Operating System* (NOS):** Fornece um sistema de ficheiros comum, partilha de impressoras, aplicações e bases de dados e a capacidade de gerir vários aspetos de uma rede.

***New Generation Workplace* (NGWP):** Nome do Programa cujo objetivo é disponibilizar uma nova solução de estação de trabalho aos colaboradores do CLIENTE.

***Out-of-box*:** Que funciona imediatamente após a instalação, sem qualquer configuração ou modificação**.**

Plano de recuperação de desastres ("*Disaster Recovery Plan* (DRP)": Plano de continuidade do negócio no caso de um desastre, que destrua parte ou todos os recursos do negócio, incluindo equipamento de TI, registos e o espaço físico de uma organização**.**

***Pool* de posto de trabalho virtual:** Grupo de máquinas virtuais configuradas de modo idêntico num servidor do *Host* de Virtualização, gerido por um *hypervisor* (Monitor de Máquina Virtual).

Pontos de gestão ("*Management Points*"): Função do sistema que, num *Site*, fornece aos clientes informação sobre localização de serviços e políticas, recebendo também, dados de configuração dos clientes. Quando fazemos *deploy* de *Software* para um cliente, o mesmo envia um pedido de conteúdo para um *Management Point*. O *Management Point* envia ao cliente uma lista dos *Distribution Points* preferenciais, e o cliente usa um deles como fonte de conteúdos. Se os conteúdos não estiverem disponíveis no *Distribution Point* preferencial, o *Management Point* envia ao cliente uma lista com os *Distribution Points* que têm esse conteúdo disponível.

***Primary Site* (PS):** Em português, *Site* Primário**.** Adicionam-se para escalamento, para redundância ou tolerância a faltas, como um ponto local de conectividade para o administrador, regulação de conteúdos, razões políticas. Pode ser *standalone* e conter 100.000 clientes. Não pode ser filho de outro *Primary Site*.

PXE (*Pre eXecution Environment*), em português: Ambiente de pré-execução:
É um método de arrancar um computador cliente usando apenas a sua placa de rede. Desde que o computador esteja ligado à rede e suporte este *standard*, é possível contornar o procedimento normal de arranque (ou seja, *Power on* – BIOS – HDD/CD). Requere que a placa de rede obtenha um endereço IP local do servidor DHCP, de modo que, é necessário um ambiente com DHCP funcional.

PXE *initiated Deployments*: Permitem aos computadores cliente requisitar um *Deployment* pela rede. Neste método de *Deployment*, a imagem do Sistema Operativo e uma *boot image Windows* PE são enviados para um *Distribution Point* que está configurado para aceitar pedidos de *boot* por PXE.

***Reference Image*:** Coleção de ficheiros e pastas compactados num ficheiro *Windows Imaging Format* (WIM) que contém o Sistema operativo, atualizações, aplicações, configurações e ficheiros. A *Reference Image* representa uma instalação de referência pré-configurada com configurações e componentes comuns, independente do *Hardware* e que pode ter 3 tipos:

- *Fat images* – imagens monolíticas que contém todas as aplicações comuns da BU, *Language packs*, *hotfixes*, *Security Updates* e outros ficheiros. Podem ser úteis para reduzir o tempo e tráfego de rede necessários para implementar uma *Workstation*, mas podem levar à necessidade de criar múltiplas *Master images* em ambientes heterogéneos;
- *Thin images* – contém apenas o Sistema operativo, *hotfixes* e *Updates*, minimizando assim o número de *Master images* necessárias. Quase todas as aplicações e configurações são aplicadas em tempo de *Deployment* ou em operação. Contudo, o tempo necessário para implementar um computador é também minimizado, quando comparado com outros cenários;
- *Hybrid images* – representam um compromisso entre as abordagens *Thin* e *Fat*, contendo quase todas as aplicações *core* comuns em todas as BU's, *Language packs*, *hotfixes*, *Security Updates* e outros ficheiros.

No caso do CLIENTE, dada o ambiente dinâmico e com grande dimensão e complexidade, a melhor solução foi escolher *Hybrid images*, devido às seguintes vantagens:

- Garantir a estandardização e interoperabilidade entre os *archetypes* físicos (*Fat Client*) e virtual (HVD/*Hybrid*);
- Usar a mesma *Master reference image* e implementar dinamicamente os requisitos específicos das BU's, durante o *Deployment*;
- Simplificar a gestão e reduzir o esforço de teste e manutenção da imagem;
- Reduzir o ciclo de vida de novas *releases* da *Master reference image***.**

***Release To Manufacturing* (RTM):** É a versão de um produto de *Software* que é dado aos fabricantes para incluir nas futuras versões dos seus produtos de *Hardware*. As versões RTM são tipicamente disponibilizadas aos fabricantes antes de serem disponibilizadas ao público, para que os fabricantes possam testar e trabalhar em eventuais problemas que surjam com os seus dispositivos. A disponibilização da versão RTM não significa necessariamente que os criadores tenham trabalhado em todos os problemas do *Software*; podem existir mais versões do produto antes de este ser disponibilizado ao público.

***Runbook*:** Contém as instruções para um processo ou uma tarefa automatizada. As etapas individuais durante um *Runbook* são chamadas de atividades. Controles adicionais fornecem informações e instruções para controlar a sequência de atividades. *Runbook*s, atividades e cada controle de *Runbook* têm propriedades configuráveis.

***Self-delivery*:** Traduz-se para português como autoentrega (sem necessidade de interação humana com o fornecedor de cada serviço)**.**

***Self-service*:** Significa que o utilizador pode, unilateralmente, requerer ou dispensar capacidades de computação, conforme necessário e de forma automática. Tudo, sem necessidade de interação humana com o fornecedor de cada serviço.

***Server Based Computing* (SBC):** Prática de executar uma aplicação centralmente num servidor e apresentar a interface numa máquina cliente**.**

***Service Level Agreement* (SLA):** Em português**:** Acordo de Nível de Serviço é a parte de um contrato de serviços entre duas ou mais entidades no qual o nível da prestação de

serviço é definido formalmente. Na prática, o termo é usado no contexto de tempo de entregas de um serviço ou de um desempenho específico.

Sistema de ficheiros Distribuídos ("*Distributed File System* (DFS)"): É constituído por um sistema de ficheiros, onde múltiplos utilizadores compartilham ficheiros e recursos, tendo como objetivo gerir servidores de ficheiros e recursos de forma eficiente, mantendo-os disponíveis e seguros para os utilizadores. São de uso comum em empresas dispersas por múltiplas localizações. Ao contrário do que acontece na solução tradicional (cliente-servidor), onde os dados são guardados no servidor, neste caso os dados podem ser guardados em vários nós (computador a operar no DFS). A isto chama-se replicação e permite atingir melhores performances e fiabilidade.

Symantec Endpoint Protection* (SEP):** Desenvolvido pela *Symantec Corporation*, é um produto de segurança para soluções corporativas, composto por antivírus e *firewall*, com gestão centralizada, para servidores e *Workstations.**

***Target computer*:** Qualquer computador cliente, no qual instalamos uma imagem do sistema operativo usando *Operating System Deployment* (OSD)**.**

***Task Sequence*:** Consiste numa combinação de passos, definidos para completar uma determinada ação. As *Task Sequences* podem ser configuradas para automatizar tarefas sem necessidade da intervenção do utilizador e têm a possibilidade de operar transversalmente ao *restart* dos computadores. Cada passo de uma *Task Sequence*, executa uma tarefa específica, como validar que o computador alvo é capaz de receber uma imagem, guardar dados do utilizador numa localização segura, fazer *deploy* de uma imagem para um computador alvo e restaurar os dados guardados do utilizador. Estes passos cumprem as suas tarefas recorrendo a *Scripts*, fornecidos pelas próprias ferramentas (MDT ou SCCM) ou pela equipa. Adicionalmente, podemos agrupar esses passos em grupos, para melhor organização e controlo de erros.
As *Task Sequences* são usadas no *Deployment* de Sistemas Operativos para construir os computadores de referência, capturar uma imagem destes, migrar os dados dos utilizadores e as configurações dos computadores e fazer o *deploy* da imagem para uma coleção de computadores alvo. Também podem ser usadas para executar outras ações do *Configuration Manager*, como o *deploy* de *Packages* de *Software* ou correr comandos customizados.

***Thin client*:** Cliente projetado para ser especialmente limitado, de forma que a maioria do processamento dos dados ocorre no servidor. Embora o termo se refira usualmente a *Software*, é crescentemente usado para computadores projetados para servir de clientes, com os recursos de *Hardware* apenas necessários para aceder a serviços remotos.

Virtualização: É de um modo geral, uma combinação de engenharia de *Software* e *Hardware* que é capaz de criar máquinas virtuais (VMs), uma abstração do *Hardware* de um computador, que permite uma única máquina atuar como se fossem várias máquinas distintas**.**

***VMware ThinApp*:** Criador de aplicações portáteis e virtualizadas, da *VMware*, é uma solução de virtualização de aplicações sem agente, que isola as aplicações dos sistemas operativos subjacentes, para eliminar conflitos e otimizar o fornecimento e a gestão destas**.**

***VMware View*:** *Software* de infraestrutura de *Desktop* virtual, em inglê**s:** *Virtual Desktop Infrastruture* (VDI), da *VMware*, que corre um *Desktop* num *Thin client*, ou PC de um utilizador, a partir dos servidores num *Datacenter*. Cria uma abordagem moderna centrada no utilizador ("*user-centric*") à computação, que maximiza a liberdade do utilizador final, otimizando também o controlo por parte do TI. O *VMware View* permite ao TI simplificar e automatizar a gestão de milhares de *Desktops* e entregar de forma segura "*Desktop as a service*" aos utilizadores, a partir de uma localização central com níveis de disponibilidade e fiabilidade não conseguidas pelo PC's tradicionais. Ao entregar acesso seguro a aplicações e dados a qualquer dispositivo, quando e onde o Utilizador necessitar, o *VMware View* disponibiliza aos utilizadores o maior nível de mobilidade e flexibilidade. O *VMware View* encapsula o Sistema Operativo, aplicações, perfis, e dados dos utilizadores em camadas isoladas para melhor gestão do *Desktop* e monta os *Desktops* dinamicamente a pedido para disponibilizar aos utilizadores uma vista personalizada do seu *Desktop* individual.

***Windows Deployment services*:** É uma tecnologia de servidor para instalações de Sistemas Operativos, baseadas em rede ("*network-based Installation*"). É direcionado para fazer *deploying* remoto do *Windows* Vista, *Windows* 7, *Windows* 8, *Windows Server* 2008 e *Windows Server* 2012, mas também suporta outros Sistemas Operativos, tendo em conta que usa *disk imaging*, em particular o formato *Windows Imaging* (WIM). Está disponivel como uma *role* no *Windows Server*.

***Windows Driver Model* (WDM):** Modelo introduzido para permitir aos desenvolvedores de *Drivers* escrever *Drivers* de dispositivos que tenham código fonte compatível com todos os Sistemas Operativos *Microsoft Windows*. *Drivers Kernel-mode* que seguem as regras WDM são chamados de "*WDM Drivers"*.

***Windows Server Update Service* (WSUS):** Função ("*role*") que disponibiliza um ponto de gestão central para obtenção de *Patches* e *hotfixes* do *Microsoft Update*. Não pode ser usado sozinho numa grande infraestrutura de TI que requeira automação, pois não contém um agendador de *Updates*, sendo por esse motivo usado em conjunto com o SCCM**.**

Lista de Figuras

Figura 1 – Desenho lógico – Descrição da abordagem centrada no Utilizador (*User Centric Approach*) do NGWP - Componentes do NGWP 12

Figura 2 - Arquitetura funcional da *Master Factory* 17

Figura 3 – Arquitetura de alto nível do *Deploy* 17

Figura 4 - Gestão do *Workplace - Deployment* 18

Figura 5 - Evolução das imagens 20

Figura 6 – Fluxo de trabalho principal da *Master Factory* 21

Figura 7 – Requisitos para permitir "*Zero Touch Installation"* 26

Figura 8 - Arquitetura do WMI 32

Figura 9 - Arquitetura simplificada do *Powershell* 37

Figura 10 - Trabalhar com *Windows PowerShell Providers* 38

Figura 11 - Visão geral do processo de criação de uma imagem personalizada do *Windows* PE, recorrendo ao DISM 50

Figura 12 - Arquitetura de funcionamento das ferramentas dos fabricantes no processo de instalação de *Drivers*. 52

Figura 13 - Diagrama de atividades do processo manual de instalação dos Drivers 61

Figura 14 - Enquadramento do processo automatizado neste trabalho, na *Task Sequence* da *Master* 63

Figura 15 - O SCCM, através da *Task Sequence*, permite correr todos as restantes ferramentas utilizadas no processo, servindo de "veículo" para o mesmo. 64

Figura 16 - O SCCM, através de uma *Task Sequence*, permite correr todas as Tasks necessárias para a concretização do processo. 65

Figura 17 - *Framework* de automatização 66

Figura 18 - Solução de semi-automatização do processo de obtenção de *Drivers* - Diagrama de atividades 67

Figura 19 - Arquitetura da solução desenvolvida 68

Figura 20 - Blocos principais da *Task Sequence* 69

Lista de Tabelas

Tabela 1 - Planeamento inicial das atividades 22

Tabela 2 - Sumário das tarefas a previstas 23

Tabela 3 - Planeamento das atividades após redefinição dos objetivos 23

Tabela 4 - Calendarização final da Participação no Projeto 24

Tabela 5 - Lista parcial dos WMI *provider*s *standard* 33

Tabela 6 - Conjunto de *provider*s embutidos do *Powershell* 38

Tabela 7 - Parâmetros usados no *cmdlet New*-CM*Driver*Package 41

Tabela 8 - Parâmetros usados no *cmdlet* Get-CM*Driver*Package 42

Tabela 9 - Parâmetros usados no *cmdlet* Start-CMContentDistribution 43

Tabela 10 - Parâmetros usados no *cmdlet* Get-CMCategory 44

Tabela 11 - Parâmetros usados no *cmdlet New*-CMCategory 44

Tabela 12 - Parâmetros usados no *cmdlet* Import-CM*Driver* 45

Tabela 13 - Opções de visualização aplicáveis a todos os comandos do *Signtool* ... 55

Tabela 14 - Tipos de *DUMP File* possíveis de obter do *Driver Verifier* (em caso de problemas) 56

Capítulo 1

Introdução

1.1 Motivação

No âmbito da disciplina de Projeto em Engenharia Informática da Faculdade de Ciências da Universidade de Lisboa, que corresponde ao último ano do Mestrado, está prevista a realização de um projeto supervisionado, que pode ser realizado numa instituição externa. Esta possibilidade é potencialmente facilitadora da integração na vida profissional, pois faz a ponte entre os ambientes académico e empresarial.

Nesse contexto, a Unisys disponibilizou uma vaga que permitiu colaborar num projeto, no âmbito de um dos diversos serviços em sistemas de informação que a empresa oferece aos seus clientes.

A Unisys é uma empresa multinacional de serviços e soluções de Tecnologia de Informação, focada na oferta de soluções seguras e rentáveis, que otimizem o desempenho do negócio dos seus clientes. Os vários projetos desenvolvidos, têm como objetivo ajudar os clientes da organização a ter maior visibilidade sobre a missão crítica do seu negócio, encontrar oportunidades, ultrapassar desafios e ter sucesso a alcançar os seus objetivos. O portefólio de soluções de negócio inclui consultoria, integração de sistemas, *outsourcing*, infraestruturas, serviços, etc. Os principais clientes onde atua são do sector público, financeiro, telecomunicações, transporte, comércio e serviços. A organização tem vários clientes e projetos sedeados em mais de cem países e utilizando diversas tecnologias e soluções.

Sabendo que a Unisys presta todos os serviços utilizando sempre as tecnologias e ferramentas de última geração, foi com interesse e motivação que decidi aproveitar esta oportunidade e mais este desafio.

1.2 Objetivos

Este relatório descreve o projeto realizado na empresa Unisys Portugal, no âmbito da disciplina de Projeto em Engenharia Informática, com o objetivo pessoal de concluir o Mestrado em Informática, na Faculdade de Ciências da Universidade de Lisboa.

No âmbito de uma das tarefas da equipa em que fui integrado, surgiu a oportunidade de automatizar a forma de obtenção dos *Drivers* a serem instalados nos computadores dos colaboradores da Organização CLIENTE, com o objetivo de permitir uma redução de custos derivados da intervenção manual desta tarefa, visto que implicava deslocações algo dispendiosas à infraestrutura do mesmo.

1.3 Contribuições

O meu trabalho foi desenvolvido no âmbito das tarefas da equipa que produz as *Master Images* dos sistemas operativos com as aplicações e *Drivers* integrados, que são depois distribuídas e instaladas nas máquinas cliente utilizadas na Organização.

O foco do meu trabalho consistiu na investigação e desenvolvimento de uma solução para automatizar, o mais possível, a forma de obtenção dos *Drivers* a disponibilizar na distribuição dessas imagens, com a pretensão de permitir uma redução de custos derivados da intervenção manual desta tarefa, visto que implicava deslocações algo dispendiosas, para testar em computadores nas condições e ambiente da infraestrutura do CLIENTE, que se localiza noutro país europeu.

A solução foi construída com base nas ferramentas, linguagens e tecnologias já utilizadas pela equipa, no desenvolvimento corrente das diversas versões das imagens, beneficiando dessa forma do conjunto das especificidades das mesmas e facilitando ainda a sua manutenção futura.

Foi ainda pensada a extensão da solução, de modo a ser possível a sua utilização noutros projetos futuramente, tendo, no entanto, ficado este ponto como consideração para possível trabalho futuro, visto não serem implementações necessárias para este projeto em particular.

1.4 Estrutura do documento

Este documento é constituído por quatro capítulos, e pretende apresentar e descrever o trabalho desenvolvido.

O segundo capítulo apresenta os objetivos do trabalho, o contexto subjacente e o planeamento efetuado para o concretizar. É ainda apresentada uma confrontação com o plano de trabalho inicial analisando as razões dos desvios ocorridos.

O terceiro capítulo descreve pormenorizadamente o trabalho realizado e as ferramentas usadas.

No quarto capítulo são apresentadas as conclusões, um sumário do trabalho realizado e um comentário crítico. São ainda apresentadas possibilidades de trabalho futuro, referindo o que falta fazer e o que poderá ser melhorado.

Capítulo 2

Projeto

Neste capítulo apresentam-se, o contexto do trabalho, os objetivos e o planeamento efetuado para os concretizar. É ainda apresentada uma confrontação com o plano de trabalho inicial e uma análise das razões que levaram aos desvios ocorridos.

2.1 Contexto

2.1.1 Apresentação do CLIENTE e do Projeto implementado pela Unisys

O CLIENTE é um grupo com mais de 200.000 colaboradores, distribuídos por cerca de setenta países. O mesmo desenvolve o seu negócio em várias vertentes (energia, gás natural e serviços) baseado num modelo de crescimento responsável e fornecendo soluções altamente eficientes e inovadoras nas respetivas áreas.

O contexto organizacional que se apresentava inicialmente no CLIENTE era a seguinte:

- Várias Unidades de Negócio ("*Business Units* (BU)"), dispersas por toda a Europa (área de intervenção para a Unisys);
- Algumas Unidades de Negócio dependiam dos *Datacenters* centrais, mas também da infraestrutura local por motivos de *performance* e rede, sendo necessário consolidar a infraestrutura;
- Algumas Unidades de Negócio pretendiam manter o controlo da sua infraestrutura, integrando fusões e programas de consolidação, sendo para isso necessária uma separação administrativa clara, por forma a permitir qualquer alteração futura na estrutura da Unidade;
- Número significativo de pequenas instalações com poucos utilizadores. Rede com uma configuração bastante complexa e pouco documentada, dado o número de *sites* e a sua localização, visto que as atividades do CLIENTE são

principalmente industriais e estão, por esse motivo, localizadas em zonas remotas onde, muitas vezes, não estão disponíveis as configurações mais atuais a nível de rede. Havia necessidade de a tornar mais flexível;

- Na gestão destas Unidades estavam envolvidas diversas organizações de TI.

O CLIENTE decidiu então reformular a sua infraestrutura de TI, para a transformar numa infraestrutura de última geração com uma oferta de serviços alinhada com o contexto do mercado atual[1], tendo definido a visão (técnica, organizacional e económica) e o programa de evolução, com a missão de dotar a Organização com uma infraestrutura comum que correspondesse às necessidades atuais e futuras, baseado em cinco programas:

- ***New Generation Workplace* (NGWP)**;
- *European Datacenter Infrastructure* (EDI);
- *Network For the Group* (NFG);
- *Identity and Access Management* (IAM);
- *Unified Communications and Collaboration* (UCC).

Como resultado desta reformulação, procurava-se obter os seguintes benefícios:

- Reduzir os custos internos, através da:
  - otimização de processos;
  - simplificação da infraestrutura de comunicações e da gestão de serviços, tirando partido de parte do *Software* e *Hardware* existentes.
- Reduzir custos de propriedade e de operação;
- Melhorar as comunicações e integração de meios de comunicação dentro da organização e com os clientes e parceiros;
- Acesso em qualquer ocasião e em qualquer local a todas as funcionalidades, para trabalhadores em mobilidade;

---

[1] Os utilizadores querem aceder aos recursos empresariais a partir de qualquer dispositivo e serem produtivos em qualquer lugar e a qualquer hora. As necessidades de negócio estão a mudar à medida que as organizações lutam para manter a competitividade. Não só é necessário suportar vários dispositivos como também proteger os dados da organização e manter a conformidade ("*compliance*"). Em determinados cenários de utilização, o armazenamento dos dados no dispositivo poderá não ser ideal ou não satisfazer as suas necessidades de segurança de dados. É necessária uma solução que permita simultaneamente satisfazer as necessidades dos utilizadores e assegurar a escalabilidade e a proteção dos dados.

- Redução do *time to market*, para novos produtos, políticas e programas em desenvolvimento, *marketing* e processos de *governance*;
- Mitigar os riscos da tecnologia e evitar a obsolescência;
- Menores necessidades energéticas e emissões de carbono.

A reformulação ao nível do programa ***New Generation Workplace*** **(NGWP)** foi implementada pela Unisys e dividida por diversas equipas, distribuídas pela Europa.

A equipa na qual fui integrado, em Portugal, tem como objetivo materializar parte do plano, implementando máquinas cliente baseadas no *Windows* 7, para todas as Unidades de Negócio e oferecer dispositivos e serviços de TI altamente flexíveis, para maximizar a produtividade dos colaboradores, ajudando a transformar as infraestruturas de TI existentes (tecnologicamente heterogéneas e geograficamente dispersas) num ambiente de trabalho comum, mais flexível e ágil, de forma a facilitar a colaboração dentro da Organização e entre as suas diversas entidades[2].

O serviço NGWP disponibilizado ao utilizador inclui um ambiente de trabalho (*workplace*), acesso às aplicações da Organização e do Negócio e dados do utilizador.

O objetivo é melhorar a experiência de local de trabalho aos utilizadores, facilitando a mobilidade (escritório, casa, viagem), o que deve endereçar uma população abrangente (colaboradores, prestadores de serviços, parceiros, clientes), ser focado na disponibilidade e tempo de resposta (*self-delivery, self-service*) e permitir o uso de variados dispositivos (*PCs, Thin clients, Tablets, smartphones*).

O serviço, do ponto de vista do utilizador, é o *Deployment* do seu posto de trabalho ("*Workplace*") incluindo o provisionamento das suas aplicações de negócio/grupo e os seus dados/documentos ("*data*") e está disponível para todos os utilizadores que o CLIENTE necessita:

- **Em qualquer localização**: facilitando a mobilidade (escritório, casa, viagem…)
- **A qualquer pessoa**: servindo todos os tipos de colaboradores (funcionários, prestadores de serviços, parceiros, clientes…)
- **A qualquer hora**: focado na disponibilidade e tempo de resposta (*self-delivery, self-service*)

---

[2] Uma entidade (que pode ser igual a uma unidade de negócio) é definida como parte da organização e é caracterizada da seguinte forma: Todos os seus utilizadores pertencem à mesma unidade de negócio e tem critérios de segurança e *templates* de *reporting* idênticos.

- **Em qualquer dispositivo**: permitindo a utilização de vários dispositivos (PC, *Thin client*, *Tablet*, *smartphone*…).

Para otimizar a *performance* e ampliar as funcionalidades do serviço, foi implementada uma estratégia de Camadas independentes:

- Sistema Operativo;
- Camada de Aplicações;
- Camada de Dados do Utilizador.

Como estas três camadas são independentes, os utilizadores podem aceder aos seus dados e usar as suas aplicações independentemente do dispositivo e versão da *Master* do Sistema Operativo.

Esta separação de camadas permite otimizar o consumo de recursos do *Backbone* no aprovisionamento de Postos de trabalho virtuais ("*Virtual Workplaces*"). De acordo com o seu *Service Level Agreement* (SLA), um utilizador pode ter acesso a um *Virtual Workplace:*

- Dedicado (persistente). A informação e aplicações do utilizador são mantidas;
- *Floating* (volátil). Com *pools* de *floating virtual Workplaces*, as máquinas virtuais são aprovisionadas a pedido ("*on demand*") e destruídas quando libertadas ("*released*").

As aplicações são geridas centralmente e virtualizadas tanto quanto possível. Ao usar o *Provisioning & Management Portal* (PMP)[3], o utilizador pode escolher que aplicações pretende, e as mesmas serão entregues automaticamente nos seus postos de trabalho. As aplicações estão associadas ao perfil do utilizador, logo são sincronizadas (instaladas/desinstaladas) em todos os *archetypes* do utilizador.

A Camada de Dados do Utilizador contém os ficheiros pessoais e perfil (personalização do ambiente de trabalho, sistema e aplicações) do utilizador. O perfil do Utilizador é aplicado nos seus postos de trabalho, e sincronizado entre eles. Os dados do Utilizador podem ser acedidos através dos seus Postos de trabalho ou diretamente a partir de dispositivos *unmanaged* [4] usados pelo mesmo.

---

[3] Portal que oferece uma interface única para o aprovisionamento de todos os dispositivos e serve como *frontend* para todos os serviços.

[4] Um dispositivo é considerado *managed* se os conteúdos são manipulados por uma entidade da Organização, em conformidade com o NGWP. Todos os outros dispositivos são considerados *unmanaged* pelo NGWP *Backbone*.

2.1.2 Requisitos do Projeto implementado pela Unisys

Os Requisitos técnicos (alto nível) a satisfazer são os seguintes:

- Acessibilidade em qualquer lugar, a qualquer pessoa, em qualquer momento e a partir de qualquer dispositivo;
- Solução integrada totalmente gerida remotamente, pela Unisys em *Datacenters* Globais e Regionais;
- Solução redundante com Plano de recuperação de desastres ("*Disaster Recovery Plan* (DRP)") integrado e alta disponibilidade;
- Manutenção dos diferentes perfis de utilizadores com os *archetypes* definidos;
- Disponibilização de capacidades multimédia a todos os *archetypes*;
- Fornecimento de armazenamento aos *archetypes*;
- Disponibilização de filas de impressão e compatibilidade de impressoras a todos os utilizadores;
- Garantia de cumprimento de todas as políticas de segurança do CLIENTE;
- Garantia de escalabilidade para fazer face às variações de utilizadores, perfis e *archetypes*;
- Disponibilização de um Portal que faz a interface entre os utilizadores e a solução e também entre os utilizadores e o negócio;
- Disponibilização de um Portal *Self-service* para terceiros;
- Disponibilização de uma arquitetura de infraestrutura organizacional que possibilita relatórios detalhados e precisos e Gestão de contratos;
- Satisfazer a interface com a CMDB do CLIENTE e possibilitar à Unisys total capacidade para gerir a solução global;
- Solução progressiva e que integra as últimas tendências de TI;
- Integração de Gestão de incidentes e problemas.

2.1.3 Migração para a solução implementada

A migração para o NGWP foi efetuada em duas fases principais (que na prática podem acontecer em paralelo):

- **Transição** que pode ser definida como o processo de planeamento, direção, execução, monitorização, controlo e fecho das atividades de transição para

projetos de *outsourcing* em larga escala. É a forma de um utilizador migrar para o espaço de trabalho do NGWP. Durante este passo, os utilizadores obtêm a sua nova estação de trabalho, devidamente configurada, sendo apenas necessário:

- Ligar ao ambiente NGWP;
- Transferir os dados pessoais da *Cloud* do NGWP;
- Aceder ao Portal *Self-service* pré-configurado, para confirmar a opção proposta, remover ou adicionar pedidos (sobre impressoras, aplicações, etc). No final deste passo, o utilizador está totalmente operacional no seu novo ambiente NGWP.

- **Transformação** é um processo iterativo, em que cada iteração corresponde à transformação de uma BU do CLIENTE. Compreende todas as atividades necessárias para projetar e construir um ambiente de trabalho operacional para o qual os utilizadores vão migrar, sendo composta por dois passos principais:

    1. Análise da Unidade de Negócio (Largura de banda, infraestrutura, *archetypes*, Aplicações, Dados, perfis de Utilizador, etc). Uma análise profunda do ambiente de cada BU existente é crítica para o sucesso da transformação da infraestrutura e transição dos utilizadores para a nova BU (transformada). Durante esta atividade são previamente validados os pressupostos, requisitos e informações apresentados pelo CLIENTE. Quaisquer lacunas ou questões detetadas durante esta atividade são documentadas e requerem uma análise de impacto antes da transição;
    2. Transformação da infraestrutura consiste em cinco atividades:
        - *Upgrade* da infraestrutura técnica (Rede, *Firewall*, AD, …), se necessário (a nível local e global);
        - Transformação das aplicações (*Packaging*, testes, integração e publicação no Portal);
        - Preparação das *Masters* necessárias para a BU*;*
        - Publicação dos recursos transformados (no Portal, atualização da CMDB);
        - Aceitação da Transformação.

Este processo cobre os seguintes tópicos:

- A estrutura de *Active Directory*:
  - Contas de utilizador e Grupos;
  - Contas de Estações de trabalho;
  - Políticas de segurança;
  - Controladores de domínio ("*Domain Controllers*");
  - Identificação de eventuais pontos de bloqueio.
- *Governance* e políticas de segurança;
- Estratégias de Domínio ("*Domain strategies*")[5];
- Servidores de ficheiros e impressão;
- Estação de trabalho (dados de Utilizador, perfis, aplicações, etc.)
- Serviço de resolução de nomes ("*Domain Name System* (DNS)");
- Topologia da Rede;
- Processo de distribuição;
- Administração e Gestão de Processos.

[5] Evolução prevista e suporte aos requisitos da mesma, para os quais a infraestrutura tecnológica deve estar preparada.

2.1.4 Solução implementada pela Unisys

O Projeto NGWP é constituído por diversos componentes. O trabalho da equipa incide exclusivamente no componente *Master Factory*.

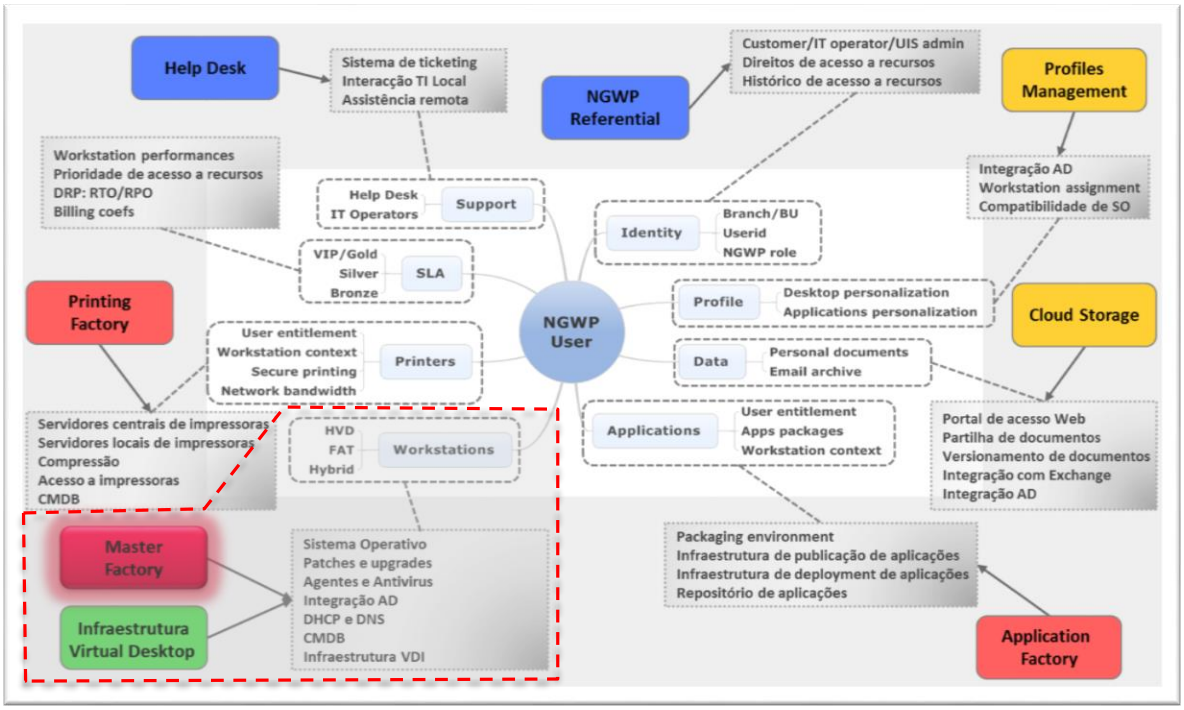


Figura 1 – Desenho lógico – Descrição da abordagem centrada no Utilizador (*User Centric Approach*) do NGWP - Componentes do NGWP

No âmbito da *Master Factory* e em termos de SCCM, a solução implementada pela Unisys, resultou numa Infraestrutura altamente complexa num cenário de Floresta de recursos ("*Resource forest*")[6], constituída por:

- 1 *Central Administration Site (CAS)* e 2 *Primary Sites (PS)*;
- 10 *Distribution Points (DP)* no *Datacenter* local e 250 *Distribution Points* remotos;
- 2 Pontos de gestão ("*Management Points*") para mediação dos pedidos ao SQL;

---

6 No modelo "*Resource forest*", é usada uma floresta separada para gerir os recursos. As Florestas de recursos não contêm contas de utilizadores, além das necessárias para administração do serviço e acesso alternativo aos recursos nessa floresta, para o caso das contas de utilizador na Floresta Organizacional se tornarem indisponíveis. As relações de confiança ("*trusts*") entre florestas são estabelecidas por forma a utilizadores de outras florestas poderem aceder aos recursos contidos na "*Resource forest*". As "*Resource forests*" fornecem isolamento do serviço, usado para proteger áreas da rede que precisem de manter um estado de alta disponibilidade.

- 6 *Windows Server Update Service*(WSUS) integrados na plataforma, para gestão de *Updates*;
- 3 *Clusters* de *SQL Server*;
- Gestão de cerca 200 mil equipamentos em 3 florestas de *Active Directory* diferentes.

Para os equipamentos a gerir foram identificados 7 tipos de utilizadores possíveis:

- Funcionários do *Call Center*;
- Funcionários da Produção;
- Funcionários da gestão de contas de clientes ("*Office-based Task workers*")
- Funcionários do *Marketing* ("*Office-based Knowledge workers*");
- Engenheiros e *Designers* ("*Power User Knowledge Workers*");
- Executivos e vendas ("*Mobile Knowledge Workers*");
- Trabalhadores temporários e externos, parceiros externos.

Tendo sido implementadas as seguintes funcionalidades:

- Distribuição de aplicações baseadas num conjunto de filtragens (Tipo de arquitetura, versão da imagem);
- Automatização da distribuição das atualizações *Windows*;
- Extensão do *Managed Object Format* (MOF)[1] para adicionar propriedades ao inventário dos computadores;
- Integração com a ferramenta *Microsoft Deployment Toolkit* 2012 SP1, tornando a solução tecnologicamente mais escalável;
- Modelo de *Master Factory* para implementação de um processo contínuo de evolução da imagem padrão, que suporta vários tipos de *archetypes* (com o objetivo de fornecer um Ambiente de trabalho adaptado aos diferentes Perfis de utilizadores):
    - *Fat Client* – Refere-se a um dispositivo físico (*Laptop*, *Desktop*, *Tablet*) a correr uma *Master Reference Image* nativa local que suporta utilização *offline* de dados e aplicações. É a forma tradicional de *Workspace* e possibilita aos utilizadores fixos (*Desktops* com boa performance, *Desktops* legados ou integração de periféricos específicos) e móveis

trabalhar com *Workstations* pertencentes à organização (permite trabalhar desligado da rede e beneficiar do *Backbone*, quando ligado);

- HVD (*Hosted Virtual Desktop*) – Estação de trabalho virtual nos modelos dedicado (guarda a informação personalizada do utilizador) e dinâmico. É baseado numa máquina virtual *VMware View* [2] a correr uma *Master Reference Image*, centralmente no *Datacenter*, que é acedida pelo utilizador através de um *endpoint*, que é um dispositivo físico suportado, a correr o *VMware View Client*, que pode ser de três tipos:
  - *Zero client*;
  - *Managed Thin client* (*Windows Embedded* ou *Linux*);
  - *unmanaged device* (*Bring Your Own Device* ou BYOD).

  Dos pontos de vista de gestão e funcionalidade, o NGWP oferece dois tipos de HVD's:

  - *Floating linked clones* - o utilizador recebe uma máquina virtual genérica (não persistente) cada vez que faz *log on*. Este tipo de HVD é adequado para utilizadores que não necessitem de personalizar o seu ambiente de trabalho e para aqueles que todas as aplicações pode ser disponibilizadas dinamicamente em tempo-real, pelos métodos de virtualização, *VMware ThinApp*[3] ou *Citrix XenApp*[4];
  - *Dedicated linked clones* - uma vez associado, o utilizador acede sempre à mesma máquina virtual, sendo este cenário adequado para utilizadores que necessitem de customização total persistente do seu ambiente de trabalho e direitos administrativos locais para instalar *Drivers* e aplicações adicionais às existentes no portal *self-service* do NGWP;

  ThinPC – Estação de trabalho que é utilizada como *Thin client* e como tal serve apenas para estabelecer a ligação aos ambientes de trabalho das estações HVD. É baseado no *Windows Thin PC*, uma versão do *Windows* 7 com funcionalidades reduzidas e portanto menos pesada, que permite utilizar máquinas antigas como *Thin clients*, para aceder a HVD's. O cliente *Thin PC* será um *endpoint* adicionado ao domínio com uma interface adaptada para focar a atenção do utilizador na ligação HVD. Por esse motivo, todos os clientes *Thin PC* são configurados para fazer *auto-logon* com uma conta local, sem direitos administrativos, que vai

lançar automaticamente a interface do cliente *VMware View* para iniciar as ligações HVD.

Uma vez que o cliente *Thin PC* estará ligado à rede corporativa do CLIENTE, seja diretamente ou via VPN, deve existir uma solução de *antivírus* local para ajudar na segurança dos recursos da rede, mesmo sabendo que o cliente *Thin PC* não mantem nenhuma informação local e estado entre *reboots*. O cliente *Thin PC* é, por esse motivo, gerido e protegido usando a maioria das tecnologias e processos dos restantes *archetypes* – implementado, gerido e atualizado pelo SCCM; protegido pelo agente *Symantec Endpoint Protection* (SEP)[5] com motores de *antivírus* e *firewall* e restrições de aplicações reforçadas tanto por *Group Policy Objects*(GPO)[6] como pelo *AppLocker*[7].

Como os dados do cliente *Thin PC* não são persistentes quando é reiniciado ou desligado (os dados não são escritos para o disco mas sim para uma *drive* RAM redirecionada), não é usado o *BitLocker*[8].

O cliente *Thin PC* tem duas versões, sendo a principal diferença entre elas o tipo de chassis e as funcionalidades disponíveis:

- *Thin PC Fix* – é baseado num chassis *Desktop* e, por definição, é considerado como um *endpoint* não amovível que está sempre nas instalações do CLIENTE, ligado diretamente à rede corporativa, via cabo de rede. A interface é reduzida ao mínimo para suportar a ligação à HVD (ou HVDs) específicos do cliente;
- *Thin PC Mobile* – é baseado num chassis *Laptop* e pode ser usado dentro ou fora das instalações do cliente, quando os utilizadores estão em trânsito, suportando ligações tanto por cabo de rede como wireless. Além da ligação aos HVDs do utilizador, a interface suporta a autenticação de ligação wireless em *hotspots wifi* públicos, como em hotéis ou aeroportos, e informa também o utilizador acerca do sinal *wifi* e estado da bateria.

o *Hybrid* – *Laptop*s ou *Desktops* físicos a correr uma máquina virtual localmente, no Sistema Operativo nativo. *Desktops* físicos que não têm uma imagem NGWP nativa, são chamados *"unmanaged"* (dispositivos, por exemplo, de prestadores de serviços, trabalhadores temporários ou externos, ou dispositivos pessoais dos funcionários).

O *archetype Hybrid* é constituído por:

- *Desktop unmanaged* + *Hypervisor* + Imagem Virtual NGWP;

- *Desktop Managed* + *Hypervisor* + Imagem Virtual NGWP.

Nota: um único utilizador pode requisitar acesso a diferentes *archetypes*, dependendo do contexto, podendo trabalhar de forma consistente, com acesso aos seus dados e configurações, independentemente do *archetype* que use.

2.1.5 *Master Factory*

A *Master Factory* tem como missão produzir e disponibilizar as várias *Master Images* necessárias.

Uma *Master Image* é o conjunto mínimo de componentes de *Software* (Sistema Operativo, *Software standard* ou do negócio, *Drivers*, agentes e portal) que tem de ser instalado numa estação de trabalho física ou virtual para permitir ao utilizador final aceder ao seu espaço de trabalho. Há vários tipos de *Masters*, que endereçam os diferentes *archetypes* que são distribuídos para os utilizadores finais.

No NGWP são usadas as soluções MDT e SCCM. O MDT (em modo autónomo) é usado para criar as *Master Reference Images*, que serão distribuídas para todos os *archetypes* em produção[7]. O SCCM em conjunto com o MDT são usados para automatizar o *Deployment* para os *archetypes* – HVD, *Fat* e *Thin PC* – configurando as especificações de cada *archetype* e *Business Unit*, nos *Datacenters* para HVD e localizações geográficas do NGWP.

Os *Deployment*s de HVD são efetuados pela equipa, nas *pools VMware Horizon View*, que estão disponíveis para os utilizadores. Os *Deployment*s de *Fat* e *Thin PC* são realizados pelas equipas locais no CLIENTE.

2.1.5.1 Funções da *Master Factory*

A *Master Factory* é responsável por criar, implementar, gerir e desenvolver as *Master reference images* para todos os *archetypes*. No âmbito da *Master Factory* estão todas as atividades e processos necessários para suportar todas as seguintes fases, para todos os *archetypes* e independentemente das restrições e requisitos específicos das Unidades de Negócio:

[7] É na remanescente fase de *Deploy* que são endereçados e adaptados à realidade de implementação de cada Unidade de Negócio

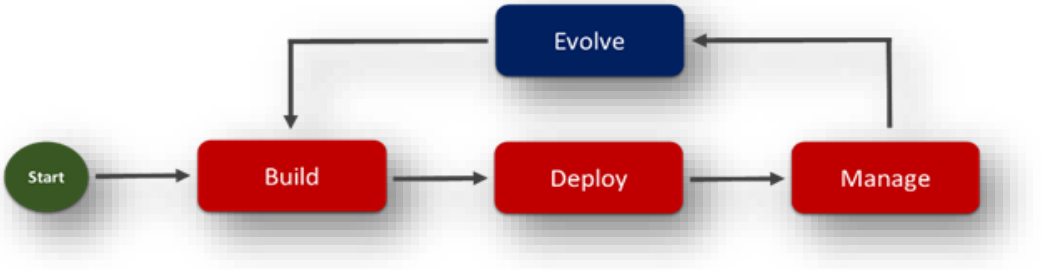


Figura 2 - Arquitetura funcional da *Master Factory*

1. ***Build*** – Criação da imagem *Master Reference Image* universal, a ser usada em todos os *archetypes*, usando um processo totalmente automatizado, minimizando-se assim a probabilidade de erro humano e garantindo a estandardização e qualidade de todas as *Master Reference Images*. O processo de *Master Build* é executado num ambiente isolado e é completamente independente de configurações específicas, como serviços de diretório, aplicações *in-house*, sistemas de gestão ou modelo de *Hardware*;

2. ***Deploy*** – Distribuição e instalação da *Master Image* em *archetypes* físicos e virtuais.

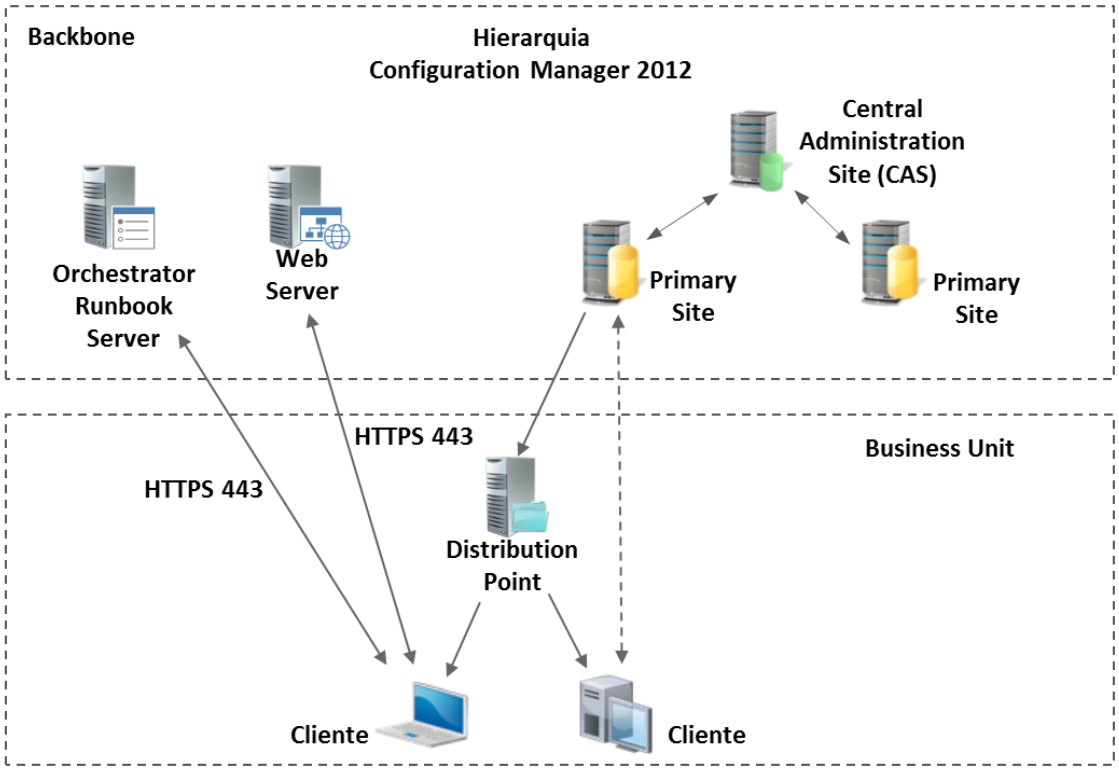


Figura 3 – Arquitetura de alto nível do *Deploy*

É durante esta fase que os *archetypes* são instalados e configurados para o ambiente NGWP e preparados para os utilizadores finais. As soluções de *Deployment*, cenários e métodos variam de acordo com cada *archetype* mas, mais uma vez, e para garantir o processo de automação e estandardização, todos seguem o mesmo *workflow* lógico de *Deployment*.

Notas: Conforme a implementação de *Active Directory* de cada unidade de negócio, pode ser necessário criar *Web Servers* ou *Orchestrator Runbook Servers* adicionais para diferentes localizações físicas.

Para localizações remotas sem *Distribution Points* locais, os clientes obtêm os conteúdos diretamente do *Primary Site*.

3. ***Manage*** – Inventário e controlo, gestão de *patch*, ciclo de vida e suporte aplicacional para cada *archetype*;
4. ***Evolve*** – Atualização das *Master Reference Images* com novos sistemas operativos, aplicações ou configurações, de acordo com os princípios da gestão do ciclo de vida do NGWP.

A ***Gestão do Workplace*** pode ser separada em três fases principais:

1. Deployment

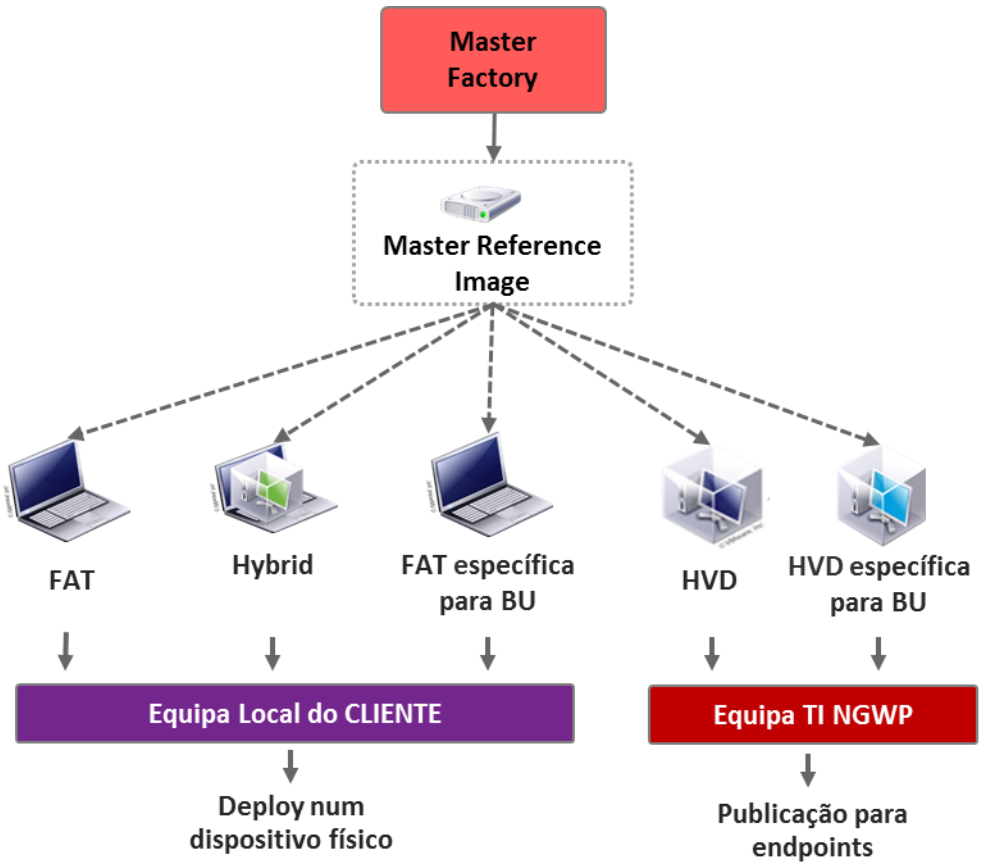


Figura 4 - Gestão do *Workplace* - *Deployment*

- Instalação e reinstalação físicas, efetuadas pela equipa local do CLIENTE, dos *archetypes Fat* e *Hybrid*;
- Ambos os HVD (*floating* e *dedicated linked clones*) são publicados para os *endpoint*s, após terem sido requisitados através do portal *self-service*;
- A configuração da ligação para *unmanaged endpoints* para os *archetypes* HVD é feita pela equipa, no local;
- Aplicações comuns para cada Unidade de Negócio são disponibilizadas de forma instantânea aos *endpoint*s.

2. Operation

- Toda a gestão de *patch* e atualização de agentes para as *pools* dedicadas dos *archetypes Fat*, HVD e *Hybrid* será efetuada pelo SCCM;
- Os *archetypes* HVD *floating* serão atualizados *offline*, através de um WSUS dedicado, pela recomposição da *Pool Master template* da *View* e usando-o nos *refresh cycles* subsequentes da *Pool*;
- Uma vez que todas as aplicações serão requisitadas através do portal *self-service* ou *deployed* durante a fase de *Deployment* não haverá atualizações de aplicações no âmbito da Gestão do *workplace*;
- A gestão do *Antivírus* é feita através do *Symantec Endpoint Protection* (SEP).
- A gestão de *patch* inclui as classificações:
    - Atualizações críticas;
    - Atualizações de Definições (para o *Endpoint Protection* usado na infraestrutura de servidor);
    - Atualizações de segurança;
    - *Update Rollups*[8];
    - Outras atualizações importantes, não incluídas no Catálogo *Default* de atualizações do *Windows*.
- A equipa de TI do NGWP aprova as atualizações a serem obtidas do *Microsoft Update Catalog* e *deployed* para os ambientes de teste;
- Os testes são efetuados ao longo de uma semana de trabalho completa;

---

[8] Pacotes de atualizações cumulativas da *Microsoft.*

- Depois de validado o sucesso, as atualizações são transferidas e distribuídas para os *archetypes* em produção e recompostas nas *View floating pools*;
- O *patching* de agentes segue o mesmo *workflow* mas sendo a equipa de TI do NGWP a criar o novo agente, em vez de usar o *Software Update Point* (SUP);
- A conformidade do *Deployment* de atualizações de Sistema Operativo e agentes é monitorizada centralmente pela equipa de TI do NGWP;
- As novas versões são então submetidas para testes da Qualidade e *Application Factory* antes de serem distribuídas pela infraestrutura;
- As novas *Master*s são distribuídas para todas as novas instalações de HVD *dedicated pools*, *archetypes Hybrid* e *Fat* e recompostas nas *View floating pools*.

3. Evolution

Funciona em ciclos, em que as atividades se repetem para cada nova *release*, que é necessário entregar ao cliente:

- Novas imagens, designadas por *Master Reference Images* são criadas quando um número significativo de *Updates*/*hotfixes* são lançados e/ou os agentes NGWP são atualizados.
- As novas versões são então submetidas às equipas de Qualidade e *Application Factory*, para testes, antes de serem distribuídas pela infraestrutura;
- As novas *Master*s são implementadas para todas as novas instalações de *pools* dedicadas HVD, *Hybrid* e *Fat* e recompostas nas *View floating pools.*

Figura 5 - Evolução das imagens

A metodologia é portanto modular, garantindo que todas as configurações e *baselines* dos *archetypes* são o mais estandardizadas possível, enquanto se reduz

também o impacto de introdução de novas Unidades de Negócio no NGWP sem conhecer os seus requisitos à partida.

A Fase 4 (Evolve) visa descrever e identificar os critérios para criar uma nova *Master Reference Image* e não atualizar a que está em uso e, portanto, não entrou em consideração na definição desta metodologia.

Em termos de requisitos, algumas decisões só podem ser tomadas, após identificadas as limitações subsequentes:

- Identificar a infraestrutura de suporte para cada uma das localizações onde os *archetypes* serão fisicamente fornecidos e implementados;
- Identificar as aplicações e configurações que devem ser configuradas para todas as Unidades de Negócio;
- Identificar requisitos específicos de cada Unidade de Negócio.

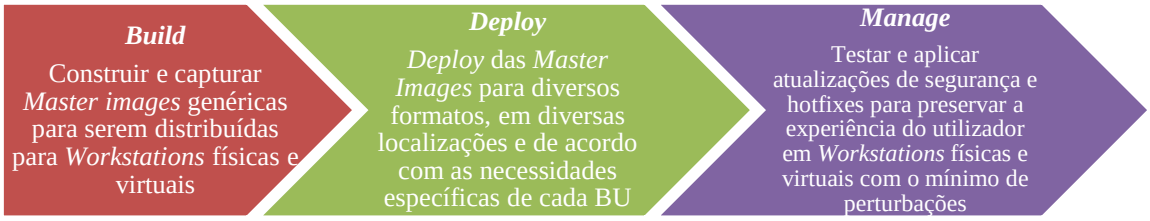


Figura 6 – Fluxo de trabalho principal da *Master Factory*

Sintetizando, as principais atividades da *Master Factory* são:

- Desenvolvimento e manutenção das *Task Sequences* e *Scripts*;
- Investigação sobre novos requisitos e resolução de defeitos;
- Qualificação de *Drivers* para as imagens *Fat* e *ThinPC*;
- Testes: Arquitetura, Instalação, Experiência do Utilizador, qualificação de *Hardware.*

2.2 Objetivos do trabalho, no contexto do Projeto apresentado e das funções da *Master Factory*

No âmbito da Qualificação de *Drivers* para as imagens *Fat* e *ThinPC* (uma das tarefas da *Master Factory*), surgiu a oportunidade de aperfeiçoar a forma de captura de *Drivers*, com o objetivo de permitir uma redução de custos derivados da intervenção manual desta tarefa, visto que implica deslocações algo dispendiosas à infraestrutura do cliente, para testar nas "Máquinas de referência" nas condições e ambiente da infraestrutura do CLIENTE, que se localiza noutro país.

As “Máquinas de referência” são usadas para capturar as diversas imagens que são posteriormente instaladas nas máquinas dos utilizadores finais.

Para concretizar esta solução foi necessário estudar a infraestrutura, ferramentas e tecnologias envolvidas, para perceber a forma como esta instalação é efetuada e encontrar uma solução viável e exequível, no contexto do projeto. Esta funciona com recurso às mesmas tecnologias e ferramentas correntemente utilizadas pela equipa, no sentido de conseguir as automatizações necessárias para permitir a instalação e captura dos *Drivers* que, após devidamente testados nas máquinas físicas de referência, são depois capturados e exportados, por forma a ficarem guardados no Repositório de *Drivers* do SCCM, para poderem então ser distribuídos para as máquinas cliente. Este processo, que é apresentado e descrito no capítulo seguinte, necessita apenas que alguém (sem necessidade de grandes conhecimentos técnicos e que poderá apenas seguir um pequeno guião, de procedimentos simples, durante o arranque das máquinas), com acesso à infraestrutura do CLIENTE, inicie essas máquinas ligadas à rede do ambiente NGWP, evitando as deslocações frequentes e dispendiosas de especialistas da equipa às instalações.

2.3 Planeamento

2.3.1 Planeamento inicial do trabalho

O planeamento inicial das atividades a desenvolver foi o seguinte:

Tabela 1 - Planeamento inicial das atividades

Atividades	**Mês**
Formação nas ferramentas e tecnologias e enquadramento no projeto	1, 2
Relatório preliminar	2
Participação no projeto	3, 4, 5, 6, 7, 8, 9
Relatório final	9

A participação no projeto teve a duração de cerca de 9 meses, mas como já era previsto, o planeamento proposto poderia ter de ser ajustado, bem como as tarefas efetivamente concretizadas, cuja previsão foi inicialmente a seguinte:

Tabela 2 - Sumário das tarefas a previstas

Etapa	Tarefas
Formação nas ferramentas e tecnologias e enquadramento no projeto	Estudo de conceitos relacionados com redes: modelo OSI, Protocolo TCP/IP, subnetting, DNS, DHCP, NAT, VPN, *routing*, etc; Estudo de Serviços de diretório/*Active Directory*; Estudo do *Windows Server* 2012; Estudo de conceitos de virtualização; Estudo do *System Center* e em particular do componente *Configuration Manager* (SCCM) 2012 Enquadramento na equipa e no contexto do projeto Enquadramento nas tarefas executadas pela equipa no projeto
Relatório Preliminar	Escrita do relatório preliminar (1 semana)
Participação no projeto	Reprodução de tarefas específicas da equipa, no contexto do projeto, em ambientes de teste Participação nas tarefas específicas da equipa, no contexto do projeto, em ambiente de produção
Relatório final	Escrita do relatório final (2 semanas)

Estaria, portanto, prevista a participação nas tarefas específicas da equipa, no contexto do projeto, mas houve uma necessidade de focar num problema mais específico, que se considerou ser útil explorar e tentar otimizar, levando a algumas alterações nas próprias atividades. Após esse ajustamento, a atividade denominada "Participação no projeto", passou a ficar planeada da seguinte forma:

Tabela 3 - Planeamento das atividades após redefinição dos objetivos

Participação no projeto	Análise do problema (1 semana) Investigação das soluções possíveis (3 semanas) Investigação e desenvolvimento de uma solução (6 semanas) Testes em ambiente de teste e eventuais correções (2 semanas) Testes em ambiente de produção (1 semana) Documentação da solução (1 semana)

Tendo previsto já nesta altura que a escrita do Relatório Final seria efetuada após o término do "Contrato de Estágio", como se verificou.

2.3.2 Análise dos desvios ao planeamento

Dado o meu desconhecimento, não só das ferramentas base em uso, como das linguagens envolvidas e havendo necessidade de aprender *Powershell* do *Zero*, entre outras linguagens, tecnologias e ferramentas, e o facto de o projeto principal ser bastante complexo e ter sido moroso de entender, agravado pelo facto de também não ter a formação base necessária na área de Sistemas de Informação, o planeamento derrapou um pouco e necessitei de mais dois meses e meio para conseguir finalizar e testar toda a análise e desenvolvimento necessários para a solução poder ser usada na prática e poder documentar a mesma da melhor forma possível, de modo que, no final, a tarefa "Participação no projeto", resultou no seguinte:

Tabela 4 - Calendarização final da Participação no Projeto

Participação no projeto	Análise do problema (1 semana) Investigação das soluções possíveis (3 semanas) Investigação e desenvolvimento de uma solução; inclui estudo de *Powershell*, WMI, *Windows Registry*, DISM e restantes ferramentas e tecnologias necessárias (15 semanas) Testes em ambiente de teste e eventuais correções (3 semanas) Testes em ambiente de produção (1 semana) Documentação da solução (1 semana)

Capítulo 3

Trabalho realizado

Neste capítulo são descritas as ferramentas e tecnologias usadas, o trabalho realizado e a contribuição concreta do Projeto.

3.1 Ferramentas usadas

3.1.1 *Microsoft Deployment Toolkit* (MDT)

O *Microsoft Deployment Toolkit* (MDT) é um acelerador de solução que oferece um conjunto unificado de ferramentas, processos e diretrizes para automatizar o *Deployment* de sistemas operativos, *Patches*, *Drivers* e aplicações em *Desktops* e servidores, reduzindo o tempo de *Deployment* e padronizando as imagens.

Aceleradores são conjuntos de documentos, *Scripts*, recursos auxiliares e metodologia para ajudar a desenhar, distribuir e manter as tecnologias e produtos *Microsoft*.

Benefícios do MDT:

- Acelerar e automatizar *Deployment*s de Sistemas Operativos;
- Reduzir o tempo de *Deployment* com imagens standardizadas;
- Usar deteção automática de plataforma (32 ou 64 bit) para garantir o *Deployment* dos *Drivers* corretos;
- Dar aos utilizadores a possibilidade de iniciar e personalizar o *Deployment*, com um *Wizard* fácil de usar dentro do *Microsoft System Center Configuration Manager* (SCCM);
- *Deployment*s *Zero-Touch* totalmente automatizados com recurso ao SCCM e *Windows Deployment tools*.

Com o MDT apenas conseguimos obter a *Lite Touch Installation* que é um processo dinâmico, mas ainda com alguma interação mínima durante o processo. Mas, se usarmos o MDT juntamente com o *System Center Configuration Manager* é possível implementar a *Zero Touch Installation*, um processo onde é possível instalar um sistema operativo completo sem nenhuma interação por parte do administrador.

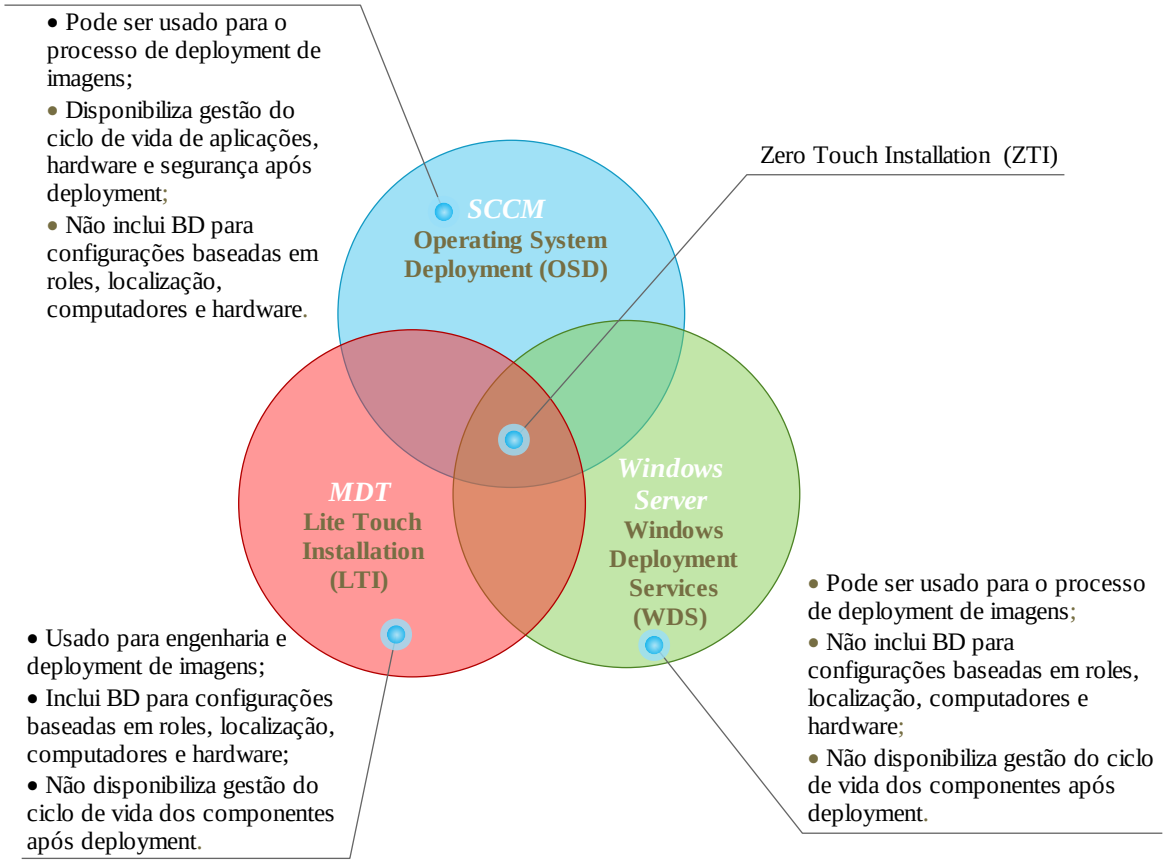


Figura 7 – Requisitos para permitir "*Zero Touch Installation*"

3.1.2 *System Center Configuration Manager* (SCCM) 2012

O *Microsoft System Center* 2012 é um pacote de produtos de gestão de servidores, totalmente preparado para *Smartphones*, *Tablets* e a *Cloud* e para gerir *clusters* de servidores de alta performance, que ajuda na configuração e manutenção destes ambientes, disponibilizando ferramentas de gestão para servidores virtualizados, infraestrutura de rede, *Cloud computing* e *Datacenters*. Permite a integração de um conjunto alargado de tecnologias numa *Cloud* privada coerente, gerir centralmente diferentes *hypervisors* a partir de um único painel de controlo, com suporte para:

- *Windows Server Hyper-V*;
- *VMware vSphere*;
- *Citrix XenServer*;

- Monitorizar o *Windows Server*;
- *Sun Solaris* e várias distribuições *Linux* e *Unix*
- Integrar conjuntos de ferramentas de HP, CA, BMC, EMC, e *VMware* nos *workflows*.

Fazem parte do *System Center* os seguintes componentes:

App Controller disponibiliza uma consola unificada que ajuda a gerir *Clouds* públicas e privadas, assim como máquinas virtuais e serviços baseados na *Cloud*, permitindo aos proprietários de aplicações criar, configurar, implementar e gerir novos serviços com facilidade.

Data Protection Manager (DPM) é uma solução de *backup* e *recovery* para *workloads Microsoft*. Disponibiliza proteção *Out-of-the-box* para ficheiros e pastas, *Exchange Server*, *SQL Server*, *Virtual Machine Manager*, *SharePoint*, *Hyper-V* e computadores cliente. Para implementações de larga escala, também permite monitorizar os *backups* através de uma consola central ou remotamente.

Endpoint Protection é construído sobre o SCCM e fornece uma infraestrutura unificada para gestão da segurança e de conformidade de clientes, ajudando a simplificar e aprimorar a proteção de *endpoints*.

Operations Manager disponibiliza diagnóstico aprofundado das aplicações e monitorização flexível da infraestrutura, que ajuda a garantir a performance e disponibilidade previsíveis das aplicações vitais e oferece uma visualização abrangente do *Datacenter* e *Clouds*, tanto privadas como públicas.

Orchestrator permite a coordenação, integração e automação dos processos, através da criação de *Runbooks*[9], que ajudam a definir e padronizar as melhores práticas e melhorar a eficiência operacional.

Service Manager fornece uma plataforma integrada para automatizar e adaptar as melhores práticas na gestão de serviços, tais como as encontradas na *Microsoft Operations Framework* (MOF) e na *Information Technology Infrastructure Library* (ITIL)[10] permitindo *self-service* flexível e processos estandardizados de *Datacenter* que podem ajudar a integrar pessoas, *workflows* e conhecimento nas aplicações e infraestrutura corporativa. Estão integrados processos para resolução de incidentes e problemas, controlo de alterações e gestão do ciclo de vida dos recursos.

Virtual Machine Manager (VMM) é uma solução de gestão para o *Datacenter* virtualizado, que permite a configuração e gestão do *Host* de virtualização, rede e

recursos de armazenamento, de modo a criar e instalar máquinas virtuais e serviços em *Clouds* privadas.

Unified Installer[9] é uma ferramenta que disponibiliza uma única interface de utilizador para instalar sete componentes do *System Center* 2012, permitindo instalação distribuída a partir de um ponto central, utilizando o programa de instalação existente de cada componente.

Configuration Manager disponibiliza uma solução abrangente para gestão de configurações e alterações, permitindo a instalação de sistemas operativos, aplicações, atualizações, monitorização e manutenção da conformidade corporativa das configurações dos dispositivos. Possibilita ainda a monitorização e o inventário de *Hardware* e *Software* e a administração remota dos dispositivos.

A nossa equipa está dedicada a administração de funcionalidades relacionadas apenas com o componente *System Center* ***Configuration Manager*** (**SCCM**) 2012, havendo outras equipas que também exploram este componente, mas apenas a nível operacional.

O SCCM é uma solução de gestão *Enterprise*, que faz parte do *Microsoft System Center Suite*, e tem a capacidade de gerir *Workstations* e servidores *end-to-end*, incluindo inventário de *Software* e *Hardware*, distribuição de aplicações, *Deployment* de sistemas operativos, gestão de *patch* e controlo de *compliance*, entre outras funcionalidades.

O SCCM 2012 permite o equilíbrio entre as exigências dos utilizadores e os requisitos de TI, focando-se em três aspetos:

- Unificar a Infraestrutura – O SCCM 2012 consolida a infraestrutura para permitir a agilização das operações, com uma infraestrutura unificada, transversal aos ambientes físico e virtual;

- Dar mais autonomia aos Utilizadores – O SCCM 2012 apresenta uma abordagem centrada no utilizador, que permite a disponibilização de aplicações e serviços aos utilizadores em qualquer parte e em qualquer dispositivo e *self-service* de aplicações através do *Application Catalog*.

---

9 Apenas para testes e avaliação.

- Simplificar a Administração – O SCCM 2012 facilita a administração dos sistemas cliente, através de uma abordagem centrada no utilizador, relacionando-o com as aplicações que utiliza. Por exemplo, quando um utilizador acede a uma aplicação a partir de um portátil da empresa, o SCCM torna esta aplicação disponível localmente, mas quando o mesmo utilizador tenta aceder à mesma aplicação a partir de um dispositivo pessoal, o SCCM percebe o contexto e faz um *stream* de uma instância criada na App-V.

3.1.2.1 Hierarquia do SCCM

Camada 1 - *Site* de Administração Central, em inglês *Central Administration Site* (CAS), deve ser instalado primeiro. Gere múltiplos Sites Primários e não tem clientes. É usado apenas para administração e *reporting*;

Camada 2 - *Site* Primário, em inglês *Primary Site* (PS), pode ser autónomo e conter 100.000 clientes. Não pode ser filho de outro *Primary Site*. Adicionam-se para escalamento, para redundância ou tolerância a faltas, como um ponto local de conectividade para o administrador, regulação de conteúdos ou políticas.

Camada 3 - *Site* Secundário, em inglês *Secondary Site* (SS), é adequado para instalar em redes de banda baixa e pode suportar até 2500 clientes. Administração baseada em regras (em inglês: *role-based*), com a preocupação de evitar o escalamento do tráfego.

Ponto de Distribuição, em inglês *Distribution Point* (DP), é adequado para instalar em redes de banda baixa e suporta até 500 clientes.

O SCCM estende e trabalha com as tecnologias e soluções existentes, da *Microsoft*:

- Usa os Serviços de Domínio ***Active Directory*** para segurança, localização de serviço, configuração e para descobrir os utilizadores e dispositivos que deseja gerir;

- Usa o *Microsoft SQL Server* como um base de dados de gestão de alterações e integra-se com o *SQL Server Reporting Services* (SSRS) para produzir relatórios para monitorizar e acompanhar as atividades de administração;

- Muitas das funções do sistema de *Site* do SCCM que fornecem a funcionalidade de administração usam os serviços da *Web* dos *Internet Information Services* (IIS);

- O Serviço de transferência inteligente em segundo plano, em inglês *Background Intelligent Transfer Service* (BITS) e a *BranchCache* [11] podem ser usados para ajudar a gerir a largura de banda disponível.

Além disso, o SCCM pode integrar com:

- *Windows Server Update Services* (WSUS);
- *Network Access Protection* (NAP);
- Serviços de Certificados;
- *Exchange Server* e *Exchange Online*;
- Política de Grupo;
- Função do Servidor DNS;
- *Kit* de Instalação Automatizada do *Windows*, em inglês: *Windows Automated Installation Kit* (AIK);
- Ferramenta de Migração de Estado do Utilizador, em inglês: *User State Migration Tool* (USMT);
- Serviços de *Deployment* do *Windows*, em inglês: *Windows Deployment Services* (WDS);
- Área de Trabalho Remota;
- Assistência Remota.

3.1.3 *Windows Management Instrumentation* (WMI)

O *Windows Management Instrumentation* (WMI)[12] é um conjunto de extensões ao *Windows Driver Model* (WDM)[13] que fornece uma interface com o Sistema Operativo, através da qual os componentes administrados disponibilizam informações e notificações. É a implementação da *Microsoft* do *Web-Based Enterprise Management* (WBEM)[14], uma iniciativa do sector para estabelecer padrões de acesso e partilha de informações de administração através de uma rede empresarial. O WMI é compatível com o WBEM e fornece suporte integrado para o *Common Information Model* (CIM)[15], o modelo de dados que descreve os objetos existentes num ambiente.

O WMI inclui um repositório de objetos compatível com CIM, que é a base de dados das definições do objeto e o gestor de objetos CIM, que trata da recolha e manipulação de objetos no repositório e reúne informações dos fornecedores de WMI.

Os fornecedores de WMI funcionam como intermediários entre o WMI e os componentes do sistema operativo, aplicações e outros sistemas. Por exemplo, o fornecedor de registos obtém informações do registo. Os fornecedores disponibilizam informações sobre os respetivos componentes e podem fornecer métodos para os

manipular, propriedades que possam ser definidas ou eventos que possam alertar para alterações nos componentes.

O WMI pode ser utilizado por ferramentas de administração de computadores e sistemas de programação ou de processamento de *Scripts* (tal como o *Windows Script Host*) para obter detalhes de configuração sobre a maioria dos aspetos dos sistemas, incluindo aplicações de servidor, ou para afetar as alterações efetuadas nos sistemas.

Um vasto conjunto de funcionalidades do *Windows* são expostos através do WMI.

Soluções de *Deployment* como o MDT e o SCCM usam o WMI para obter informação acerca do *Hardware*. O SCCM também usa o WMI para guardar configurações e obter o inventário. Pode ser usado para configurações de leitura/escrita.

O *Powershell* tem comandos que fazem a interação com o WMI.

3.1.3.1 Arquitetura do WMI

A Arquitetura do WMI consiste em três camadas principais [16]:

- *Managed resources* - Um *Managed resource* (recurso gerível) é qualquer componente lógico ou físico, que é exposto e gerível pelo uso do WMI. Recursos do *Windows* que podem ser geridos por WMI incluem: o Sistema, discos, periféricos, *logs* de eventos, ficheiros, pastas, sistemas de ficheiros, componentes de rede, impressoras, processos, configurações do registo, segurança, serviços, *shares*, *Active Directory*, *Windows Installer*, *Drivers* de dispositivos *Windows Driver Model* (WDM), etc. Um WMI *Managed resource* comunica com WMI através de um *provider*. O termo instância ("*instance*") é normalmente usado para referir uma representação virtual do *Managed resource*;
- *Consumers* - são a camada de topo. Um *consumer* pode ser um *Script*, uma aplicação de *Enterprise Management*, uma aplicação *Web-Based*, ou outra ferramenta administrativa, que aceda e controle informação de administração disponível através da infraestrutura do WMI.
- WMI *infrastructure* – é a camada intermédia da infraestrutura do WMI e consiste em três componentes principais:
    - *Common Information Model Object Manager* (CIMOM);
    - *Common Information Model* (CIM) *repository*;
    - *Provider*s.

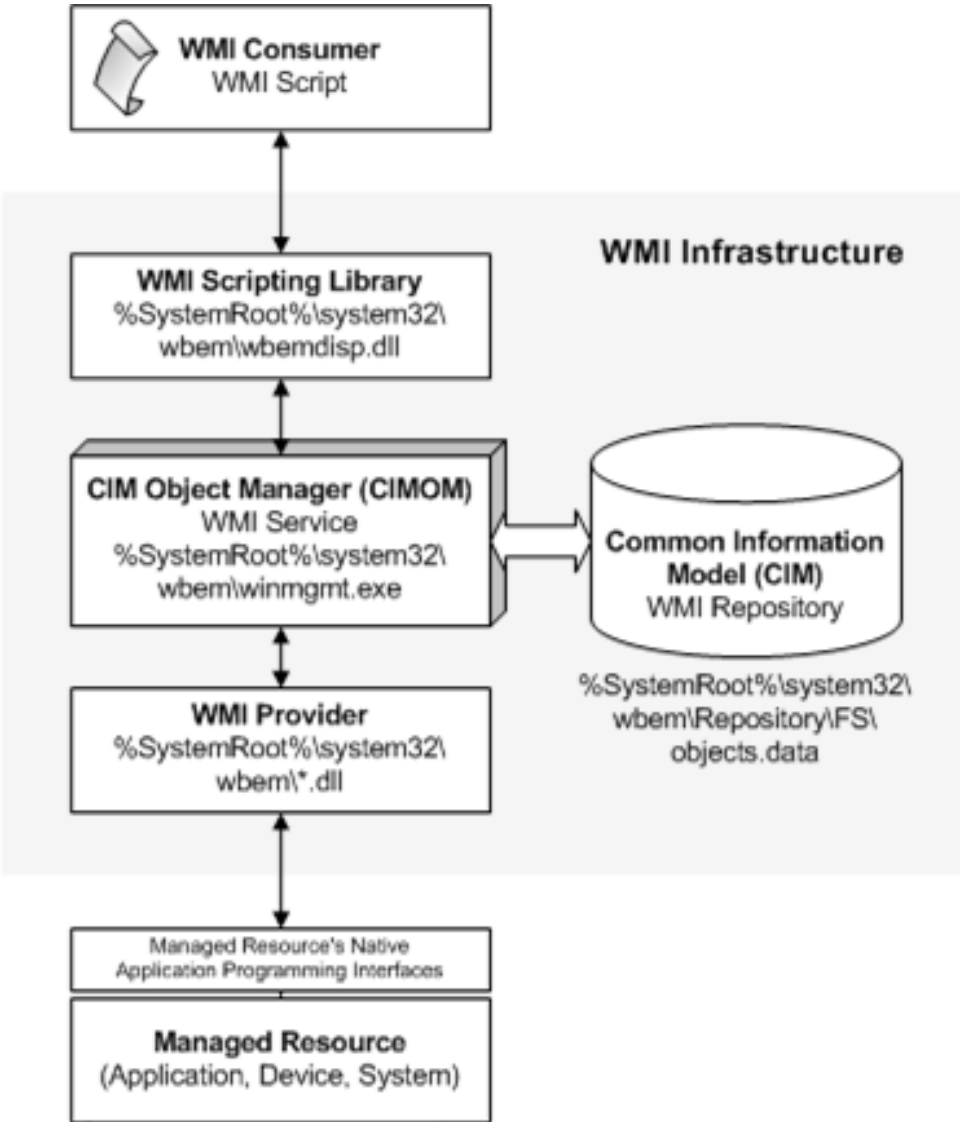


Figura 8 - Arquitetura do WMI

Juntos, fornecem a infraestrutura através da qual a informação de configuração e administração é definida, exposta, acedida e encontrada.

Um quarto componente, embora pequeno, mas absolutamente essencial para efeitos de *scripting* é a WMI *scripting library*. A WMI *scripting library* fornece um conjunto de objetos de automação através dos quais as linguagem de *scripting*, acedem à infaestrutura do WMI e é implementada num único componente de automação: wbemdisp.dll, que reside fisicamente na pasta: *Systemroot**System*32\Wbem [17].

WMI *Provider*s

Os WMI *provider*s actuam como intermediários entre o WMI e um *Managed resource*. Os *Provider*s solicitam informação de, e enviam instruções para WMI *Managed resources* em nome das aplicações e *Scripts* consumidores. Os *Provider*s

escondem os detalhes de implementação únicos para um *Managed resource*, expondo o *Managed resource* à infraestrutura do WMI com base nos standards WMI, *uniform access Model*. Os WMI *provider*s comunicam com os seus respetivos *Managed resources* usando as APIs nativas dos *Managed resources*, e comunicam com o CIMOM usando as interfaces programáticas do WMI. Por exemplo, o *provider Event Log* integrado, chama as APIs Win32 *Event Log* para aceder a *Event logs*.

Graças à arquitetura extensível do WMI, os desenvolvedores de *Software* podem desenvolver e integrar *provider*s adicionais para expor as funções administrativas específicas aos seus produtos.

Os *Provider*s são geralmente implementados como *dynamic link libraries* (DLLs) e residem na pasta: %*SystemRoot*%*System*32\wbem. O WMI inclui muitos *provider*s integrados para a família de Sistemas Operativos *Windows*. Estes *provider*s, também conhecidos como *standard provider*s, oferecem informação e funções administrativas de fontes conhecidas dos Sistemas Operativos, tais como o Win32 *subSystem*, *Event logs*, performance *counters*, e *registry*. A Tabela 1 lista muitos dos *standard* WMI *provider*s incluídos na família de Sistemas Operativos *Windows*.

Tabela 5 - Lista parcial dos WMI *providers standard*

Provider	**DLL**	***Namespace***	**Descrição**
Active Directory provider	dsprov.dll	root*Directory*\ldap	Mapeia objetos *Active Directory* para o WMI.
Event Log provider	ntevt.dll	root\cimv2	Gere *Event logs* do *Windows* , por exemplo, *read*, *backup*, *clear*, *copy*, *delete*, *monitor*, *rename*, *compress*, *uncompress*, e altera configurações de *Event log*.
Performance *Counter provider*	wbemperf.dll	root\cimv2	Dá acesso a informação de performance.
Registry provider	stdprov.dll	root*Default*	*Read*, *write*, *enumerate*, *monitor*, create e *delete* chaves e valores do *registry*.
WDM *provider*	wmiprov.dll	root\wmi	Fornece acesso a informação em WDM device *Drivers*.
Win32	cimwin32.dll	root\cimv2	Disponibiliza informação sobre computador, discos, periféricos,

provider			ficheiros, pastas, sistemas de ficheiros, componentes de rede, sistema operativo, impressoras, processos, segurança, serviços, *shares*, etc.
Windows Installer provider	msiprov.dll	root\cimv2	Dá acesso a informação sobre *Software* instalado.

***Common Information Model Object Manager* (CIMOM)**

O *Common Information Model Object Manager* (CIMOM) trata da interação entre *Consumers* e *provider*s. O termo vem da iniciativa *Web-Based Enterprise Management* e da especificação *Common Information Model*, mantida pela *Distributed Management Task Force* (DMTF).

Podemos pensar no CIMOM como o intermediário da informação do WMI. *To*dos os pedidos e informação no WMI fluem através do CIMOM. Nos sistemas *Windows* recentes, o serviço *Windows Management Instrumentation*, winmgmt.exe, tem a função do CIMOM, que corre sob controlo do *Generic services Host process*, svchost.exe.

Adicionalmente o CIMOM fornece os seguintes serviços *core* à infraestrutura WMI:

- *Provider registration*. Os WMI *provider*s registam a informação de localização e capacidades com o CIMOM. Esta informação é armazenada no CIM *repository*.
- *Request routing*. O CIMOM usa a informação de registo do *provider* para encaminhar os pedidos dos *Consumers* para os *provider*s apropriados.
- *Remote access*. Os *Consumers* acedem a sistemas remotos WMI-*enabled* ligando-se ao CIMOM no sistema remoto. Uma vez estabelecida a ligação, os *Consumers* podem efetuar as mesmas operações que podem ser executadas localmente.
- *Security*. O CIMOM controla o acesso aos *Managed resources* do WMI validando o *token* de acesso de cada utilizador, antes do utilizador ser autorizado a ligar ao WMI, tanto no computador local como no remoto. O WMI não substitui ou contorna a segurança disponibilizada pelo Sistema Operativo.
- *Query processing*. Permite a um *consumer* fazer *Queries* a qualquer WMI *Managed resource* usando a WMI *Query* Language (WQL). Por exemplo, podemos fazer um *Query* aos *Event logs* por todos os eventos que

correspondam a um determinado *Event* ID, que tenha ocorrido nas últimas 24 horas. O CIMOM executa a avaliação da *Query* nos casos em que os *provider*s não suportem nativamente estas operações.

- *Event processing*. Permite a um *consumer* subescrever eventos que representem uma alteração a um WMI *Managed resource*. Por exemplo, podemos subescrever um evento a indicar quando o espaço num disco lógico for inferior a um determinado valor limite. O CIMOM sonda o *Managed resource* no intervalo especificado, e gera uma notificação quando a subscrição for satisfeita.

Aplicações administrativas e *Scripts* fazem chamadas ao CIMOM para coletar informação, subescrever eventos, ou para executar alguma outra tarefa administrativa. O CIMOM obtém a informação necessária do *provider* e da classe, para responder aos pedidos dos *Consumers* do CIM. O CIMOM usa a informação obtida do CIM para entregar os pedidos dos *Consumers* ao *provider* apropriado.

CIM *Repository*

O WMI é baseado na ideia de que a informação de configuração e administração de diferentes fontes pode ser representada uniformemente com um *schema*. O CIM é o *schema*, também chamado *Object repository* ou *class store* que modela o ambiente gerido e define cada parcela de informação exposta pelo WMI. O *schema* é baseado no *standard* DMTF *Common Information Model*.

Tal como o *schema* do *Active Directory* é construído com base no conceito de uma classe, o CIM consiste em classes. Uma classe é um *blueprint* para um WMI *manageable resource*. Contudo, ao contrário das classes do *Active Directory*, que representam objetos criados e armazenados no *Directory*, as classes CIM geralmente representam recursos dinâmicos. Ou seja, instâncias de recursos não estão armazenadas no CIM, mas são obtidas dinamicamente por um *provider* com base num pedido de um *consumer*. A razão para isto é simples, o estado de operação para muitos WMI *Managed resources* muda frequentemente e consequentemente tem de ser lido a pedido para garantir que é obtida a informação mais atualizada.

Nota: O termo *repository* é de alguma forma enganador no contexto do CIM. Apesar de o CIM ser um repositório e capaz de armazenar informação estática, a sua função primordial é armazenar os *blueprints* para *Managed resources*.

Da mesma forma que as classes do *Active Directory*, as classes do CIM são organizadas hierarquicamente, onde as classes filho herdam das classes pai. A DMTF mantém o conjunto das classes base *core* e *Common* das quais os desenvolvedores de

aplicações e sistemas, derivam e criam classes extensão, específicas das aplicações ou sistemas.

As Classes são agrupadas em *namespaces*, que são grupos lógicos de classes, que representam uma área específica de administração. Por exemplo, o *namespace* root\cimv2 inclui a maioria das classes que representam recursos usualmente associados com um computador e sistema operativo.

As classes CIM consistem em propriedades e métodos. As Propriedades descrevem a configuração e estado de um WMI *Managed resource*, e métodos são funções executáveis que efetuam ações no WMI *Managed resource*.

Fisicamente, o CIM reside na pasta %*SystemRoot*%*System*32\wbem*Repository*\ e consiste nos seguintes ficheiros:

- index.btr. Ficheiro de indíce *Binary-tree* (btree);
- index.map. Ficheiro de controlo de transação;
- *Object*s.*data*. CIM *repository* onde as definições dos *Managed resources* estão armazenadas;
- *Object*s.map. Ficheiro de controlo de transação.

WMI *Scripting Library*

A WMI *scripting library* disponibiliza um conjunto de objetos de automação através dos quais as linguagens de *scripting* acedem à infraestrutura do WMI.

Os objetos de automação na WMI *scripting library* fornecem um modelo de *scripting* consistente e uniforme para a infraestrutura do WMI.

A WMI *scripting library* está implementada num único ficheiro DLL: wbemdisp.dll, que reside fisicamente na pasta: %*SystemRoot*%*System*32\wbem. A WMI *scripting library* também inclui uma biblioteca de tipos chamada: wbemdisp.tlb. Podemos usar a WMI *scripting type library* para referenciar constantes WMI a partir de ficheiros baseados em XML *Windows Script* e *Scripts* WSH com a extensão .wsf.

3.1.4 *Powershell*

A administração de sistemas atualmente implica uma substancial perda de tempo com tarefas repetitivas.

O *Powershell* é uma *Framework Microsoft* para automação e configuração de tarefas que consiste numa *shell* de linha de comandos e uma linguagem de *scripting* associada, construída com base na .NET *Framework*, tendo acesso a todos os objetos COM e acesso direto ao WMI. Esta é uma linguagem poderosa, orientada a objetos, que pode ser usada para escrever e executar *Scripts* que permitem eliminar as tarefas

repetitivas e adicionar a lógica necessária para realizar atividades complexas. Suporta a programação comum, mas é menos complicada que outras linguagens de *scripting* e usa menos linhas de código nas operações.

A arquitetura simplificada do *Powershell* é a seguinte [18]:

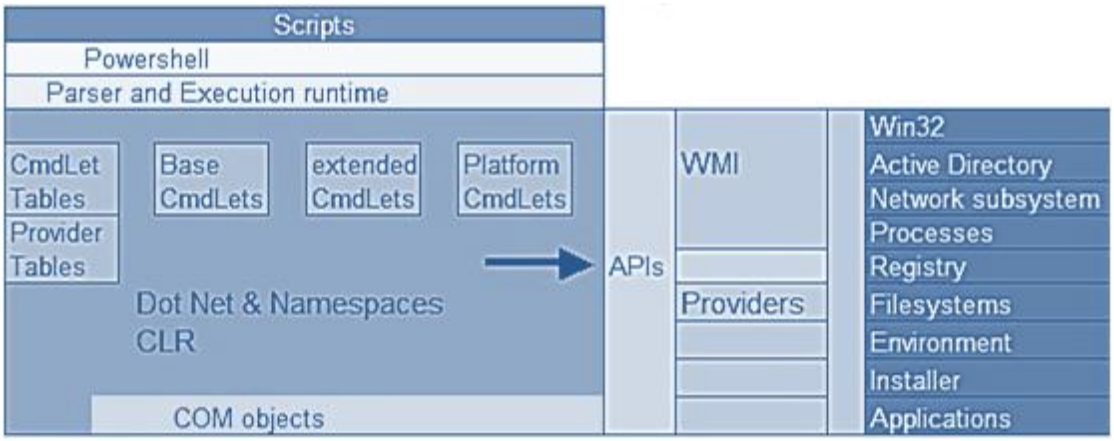


Figura 9 - Arquitetura simplificada do Powershell

O *Powershell* permite executar tarefas administrativas tanto em sistemas *Windows* locais como remotos. Desde a saída do WinPE 4.0, o *Powershell* também pode ser adicionado às *boot images* usadas no *Deployment* de Sistemas Operativos. Estas tarefas administrativas são geralmente executadas por *cmdlets* (pronuncia-se *command-lets*).

Os *cmdlets* são comandos especializados que implementam funções específicas. São os comandos nativos do *Powershell* e seguem um padrão de nome "Verbo-Substantivo" ("*Verb-Noun*"), como por exemplo Get-ChildItem, tornando-os auto-descritivos. O *output* dos *Cmdlets* são objetos, ou *collections* destes (incluindo *arrays*), e podem opcionalmente receber *input* nessa forma, tornando-os adequados para uso como recipientes numa *pipeline*.

Enquanto o *Powershell* permite que *arrays* e outras *collections* de objetos sejam escritas para a *pipeline*, os *cmdlets* processam sempre os objetos individualmente. Para *collections* de objetos, o *Powershell* invoca o *cmdlet* em cada objeto da *collection*, sequencialmente.

Podemos criar os nossos próprios *cmdlets* e usar cada um deles separadamente.

O *Powershell* pode executar quatro tipos de comandos:

- *Cmdlets*, que são programas .NET desenhados para interagir com o *Powershell*;
- *Scripts Powershell* (com sufixo .ps1);
- Funções *Powershell*;
- Programas executáveis autónomos.

Se um comando é um Programa executável autónomo, o *Powershell* lança-o num processo separado; se é um *cmdlet*, é executado no processo do *Powershell*.

O *Powershell* também permite usar esses vários comandos em combinação, para executar tarefas complexas. Conjuntos de *cmdlets* podem ser combinados em *Scripts*, executáveis, ou pela instanciação de classes .NET (ou objetos WMI/COM)[19]. Isto funciona acedendo aos dados em diferentes fontes, como o sistema de ficheiros ou registo, que são disponibilizados ao *runtime* do *Powershell* via *Windows Powershell provider*s.

Um *Powershell provider* é um componente de *Software* que se situa entre a *data store* e os *cmdlets standard*. Os *cmdlets* obtêm a informação relevante do *provider* e apresentam-na [20]:

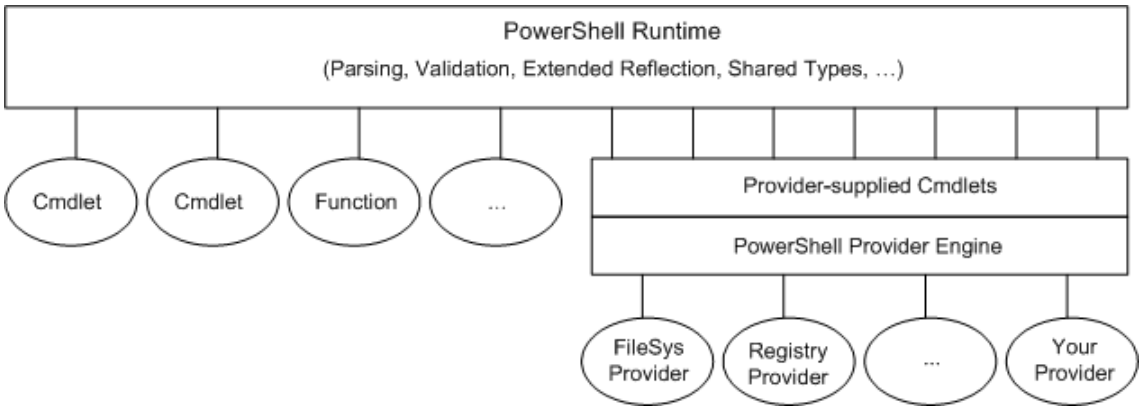


Figura 10 - Trabalhar com *Windows PowerShell Providers*

O *Powershell* inclui um conjunto de *provider*s embutidos que podem ser usados para aceder aos diferentes tipos de *data stores* e que se apresentam na seguinte tabela:

Tabela 6 - Conjunto de *provider*s embutidos do *Powershell*

Provider	**Descrição**
Alias Provider	Fornece acesso a *Windows Powershell aliases* e seus valores.
Certificate Provider	Fornece acesso de leitura às *stores* de certificados X509 e certificados.
Environment Provider	Fornece acesso às variáveis de ambiente do *Windows*.
File*System Provider*	Fornece acesso a ficheiros e pastas.
Function Provider	Fornece acesso às funções definidas no *Powershell*.
Registry Provider	Fornece acesso às chaves e valores do *System registry*.
Variable *Provider*	Fornece acesso às variáveis do *Powershell* e seus valores.
WSMan *Provider*	Fornece acesso à informação de configuração do WSMan.

Também podemos criar os nossos *Powershell provider*s e instalar *provider*s desenvolvidos por terceiros.

O *Powershell* tem tipagem dinâmica[10] e pode implementar operações complexas usando *cmdlets* imperativamente. Suporta variáveis, funções, ramificações/condições ("*branching*") (*if-then-else*), *loops* (*while, do, for, and foreach*), tratamento estruturado de erros/excepções, assim como integração com .NET.

As variáveis têm nomes que começam com $; estas podem receber qualquer valor, incluindo o *output* de *cmdlets*.

O *Powershell* tem variáveis especiais, como:

- $args: é um *array* de todos os argumentos da linha de comando passados a uma função;
- $_: refere-se ao objeto corrente da *pipeline*;
- $Env:*Path*: Caminho de ambiente para o ficheiro ou pasta [21].

As *Strings* podem estar envoltas em aspas simples ou duplas: quando em aspas duplas, as variáveis são expandidas[11].

O *Powershell* suporta parâmetros de nome e posição. Um parâmetro de nome ("*named parameter*") requere que se escreva o nome do parâmetro e argumento na chamada do *cmdlet*. Um parâmetro posicional requer apenas que os argumentos sejam escritos por ordem. Por defeito, todos os parâmetros dos *cmdlet* são "*named Parameters*" [22]. Na execução de um *cmdlet*, o trabalho de associar o valor do argumento ao parâmetro é feito pelo próprio *Powershell*, mas no caso de executáveis externos, é feito o *parsing*[12] dos argumentos pelo executável externo independentemente da interpretação do *Powershell*.

O *Powershell* permite ainda chamar qualquer método .NET através do seu respetivo nome envolto em parêntesis retos ([]), usando depois um par de "dois pontos" (::) para indicar o método estático. Exemplo: [*System*.Console]::WriteLine("*Powershell*").

---

[10] Faz verificação de tipos em run-time em vez de Compile-time.

[11] É passado o seu conteúdo.

[12] A análise sintática transforma um texto na entrada numa estrutura de dados, em geral uma árvore, o que é conveniente para processamento posterior, e captura a hierarquia implícita desta entrada. Através da análise léxica é obtido um grupo de tokens, para que o analisador sintático use um conjunto de regras para construir uma árvore sintática da estrutura. Um *token* é segmento de texto ou símbolo que fornece um significado ao texto.

Pipeline

O *Powershell* implementa o conceito de *pipeline*, o que possibilita receber o *output* de um *cmdlet* como *input* de outro. Esta é usada para construir comandos complexos, usando o operador | para ligar etapas.

Estas etapas executam em *runtime* do *Powershell*, sendo passados entre fases objetos .NET estruturados. Desta forma é eliminada a necessidade de serializar[13] as estruturas de dados ou extrai-las fazendo explicitamente *parsing* de *output* de texto.

Um objeto pode ainda encapsular certas funções que trabalham sobre os dados nele contidos, as quais se tornam disponíveis para uso pelo comando recetor. Os membros dos Objetos podem ser acedidos usando a notação . , como na sintaxe do C#.

No último comando da *pipeline*, o *Powershell* canaliza automaticamente o seu objeto de *output* para o *cmdlet Out-Default*, o qual transforma os objetos numa *stream*[14] de objetos de formato e depois os renderiza para o ecrã [23][24].

Tratamento de erros

Para tratamento de erros, o *Powershell* tem um mecanismo de tratamento de excepções baseado em .NET. Em caso de erros, os objetos que contêm informação acerca do erro (*Exception Object*s) são lançados, sendo capturados usando o *construct* "try ... catch" (ou "trap"). Em caso de erro o *Powershell* pode ser configurado para continuar a execução, sem lançar realmente a excepção.

Windows Powershell Execution Policies

O *Windows Powershell* suporta certas políticas de execução ("*Execution* policies") [25] que definem as restrições sobre as quais os ficheiros são carregados para configuração e execução.

Antes de conseguirmos executar *Scripts* numa máquina, é necessário alterar a política de execução por defeito ("*Restricted*") do *Powershell*. Isto pode ser feito a nível do Sistema, Utilizador ou sessão. A opção usada no projeto foi:

Bypass	Permite contornar temporariamente a *Execution Policy* para executarmos o que for necessário.

[13] É o processo de armazenar um objeto, incluindo todos os atributos públicos e privados para um stream, de forma a que possa ser guardado persistentemente. É nada mais do que colocar os valores que o objeto está a utilizar juntamente com suas propriedades, de uma forma que fique em série (sequencial).

[14] Corrente ou fluxo de informação.

	Ao criar um *package* com este comando, podemos chamar o *Script* sem ter de alterar a *Execution Policy* na máquina, permitindo ao Sistema permanecer inalterado e menor risco de causar um problema maior como pode acontecer ao alterar arbitrariamente a *Policy*. A responsabilidade da segurança fica totalmente a cargo do utilizador.

Sintaxe para iniciar o *Powershell* da *command line* de outra ferramenta, como Cmd.exe [26]:

```
Powershell[.exe]
       [-EncodedCommand <Base64EncodedCommand>]
       [-ExecutionPolicy <ExecutionPolicy>]
       [-InputFormat {Text | XML}]
       [-NoExit]
       [-NoLogo]
       [-NonInteractive]
       [-NoProfile]
       [-OutputFormat {Text | XML}]
       [-PSConsoleFile <File> | -Version <version>]
       [-Sta]
       [-Windowstyle <style>]
       [-File <filePath> <args>]
       [-Command { - | <Script-block> [-args <arg-array>]
                     | <String> [<CommandParâmetros>] } ]
```

3.1.4.1 *Cmdlets Powershell* específicos para o SCCM

A *Library* de *cmdlets* do *System Center Configuration Manager* permite administrar uma hierarquia do *Configuration Manager*, usando *cmdlets* e *Scripts Powershell*.

Os *cmdlets* desta *Library*, usados no trabalho, foram:

3.1.4.1.1 New-CMDriverPackage

Para criar um *Package* para cada modelo:

Sintaxe [27]:

```
New-CMDriverPackage -Name <String> -PackageSourceType <PackageSourceTypes>
{StorageCompress | StorageDirect | StorageLocal | StorageNeedsSpecifying |
StorageNOSource} -Path <String> [-Description <String>] [-Confirm] [-WhatIf]
[ <CommonParameters>]
```

Tabela 7 - Parâmetros usados no *cmdlet New*-CM*Driver*Package

Parâmetros usados	**Descrição**
-Description<*String*>	Descreve um *Driver package*.
-Name<*String*>	Especifica um nome para um *Driver package*.
-PackageSourceType	Especifica o método de leitura dos ficheiros fonte do *package*. Os

<PackageSourceTypes>	valores válidos são: -- StorageCompress -- StorageDirect -- StorageLocal -- StorageNeedsSpecifying -- StorageNOSource
-*Path*<*String*>	Especifica o caminho para a localização onde o *Configuration Manager* guarda a versão comprimida dos ficheiros fonte no *Site Server*.
<*CommonParameters*>	Este *cmdlet* suporta os parâmetros comuns: -*Verbose*, -*Debug*, -ErrorAction, -ErrorVariable, -*OutBuffer*, -OutVariable.

3.1.4.1.2 Get-CMDriverPackage

Para obter *Driver Packages*:

Sintaxe [28]:

```
Parameter Set: SearchByName
Get-CMDriverPackage [-Name <String> ] [-SecuredScopeNames <String> ] [
<CommonParameters>]

Parameter Set: SearchByIdMandatory
Get-CMDriverPackage -Id <String[]> [-SecuredScopeNames <String> ] [
<CommonParameters>]
```

Tabela 8 - Parâmetros usados no *cmdlet* Get-CM*Driver*Package

Parâmetros usados	**Descrição**
-Id<*String*[]>	Especifica um *array* de identificadores for a *Driver package*.
-Name<*String*>	Especifica um nome para um *Driver package*.
-SecuredScopeNames<*String*>	Especifica os nomes dos *Security* scopes. Um nome de *Security scope* pode ser *Default* ou criado para o efeito.
<*CommonParameters*>	Este *cmdlet* suporta os parâmetros comuns: -*Verbose*, -*Debug*, -ErrorAction, -ErrorVariable, -*OutBuffer*, -OutVariable.

3.1.4.1.3 Start-CMContentDistribution

Para distribuir o *Driver Package* criado para o(s) *Distribution Point*(s). Este *cmdlet* copia conteúdo da *content library* no *Site Server* para a *content library* nos *Distribution points*. Podemos usar este *cmdlet* para distribuir vários tipos de conteúdos, incluindo *application Deployment types*, *Packages*, *Deployment Packages*, *Driver Packages*,

Operating System images, *Operating System Installer*s, *boot images*, e *Task Sequences*. Podemos distribuir o conteúdo para *Distribution points*, *Distribution Point groups*, ou *collections* associadas com *Distribution Point groups*.

Sintaxe [29]:

```
Parameter Set: SearchByNameMandatory_DriverPackage
Start-CMContentDistribution -DriverPackageName <String[]> [-CollectionName
<String> ] [-DistributionPointGroupName <String> ] [-DistributionPointName
<String> ] [-Confirm] [-WhatIf] [ <CommonParameters>]

Parameter Set: SearchByValueMandatory_DriverPackage
Start-CMContentDistribution -DriverPackage <IResultObject> [-CollectionName
<String> ] [-DistributionPointGroupName <String> ] [-DistributionPointName
<String> ] [-Confirm] [-WhatIf] [ <CommonParameters>]
```

Tabela 9 - Parâmetros usados no *cmdlet* Start-CMContentDistribution

Parâmetros usados	Descrição
-DistributionPointName<String>	Especifica o nome de um *Distribution Point* que está associado com o *Deployment package*.
-DriverPackage<IResultObject>	Especifica um objeto *Driver package*. Para obter o objeto CM*Driver*Package, usa-se o *cmdlet* Get-CM*Driver*Package.
-DriverPackageId<String[]>	Especifica um *array* de Ids de *Driver Packages*.
-DriverPackageName<String[]>	Especifica um *array* de nomes de *Driver Packages*.
<CommonParameters>	Este *cmdlet* suporta os parâmetros comuns: *-Verbose*, *-Debug*, -ErrorAction, -ErrorVariable, *-OutBuffer*, -OutVariable.

3.1.4.1.4 Get-CMCategory

Obtém as categorias de configuração no *Configuration Manager*. As categorias de configuração oferecem um método opcional de ordenar e filtrar as configurações base e itens de configuração no *System Center Configuration Manager* e *Configuration Manager reports*.

Sintaxe [30]:

```
Parameter Set: GetCategoryByName

Get-CMCategory [-CategoryType <CategoryType> {UserCategories |
BaselineCategories | DriverCategories | AppCategories | GlobalCondition |
CatalogCategories} ] [-Name <String[]> ] [ <CommonParameters>]

Parameter Set: GetCategoryById Get-CMCategory [-Id <String[]> ] [
<CommonParameters>]
```

Tabela 10 - Parâmetros usados no *cmdlet* Get-CMCategory

Parâmetros usados	Descrição
-*CategoryType<CategoryType>*	Especifica um tipo de categoria. Os valores válidos são: - *BaselineCategories* - *DriverCategories* - *AppCategories* - *GlobalCondition* - *CatalogCategories*
-*Id<String[]>*	Especifica um *array* de IDs de categorias de configuração.
-*Name<String[]>*	Especifica um *array* de nomes de categorias de configuração.
<*CommonParameters*>	Este *cmdlet* suporta os parâmetros comuns: -*Verbose*, -*Debug*, -*ErrorAction*, -*ErrorVariable*, -*OutBuffer*, -*OutVariable*.

3.1.4.1.5 New-CMCategory

Cria uma categoria de configuração no *Configuration Manager*.

Sintaxe [31]:

```
Parameter Set: NewCategory

New-CMCategory -CategoryType <CategoryType> {UserCategories | 
BaselineCategories | DriverCategories | AppCategories | GlobalCondition | 
CatalogCategories} -Name <String> [-Confirm] [-WhatIf] [ <CommonParameters>]
```

Tabela 11 - Parâmetros usados no *cmdlet New*-CMCategory

Parâmetros usados	Descrição
-*CategoryType<CategoryType>*	Especifica um tipo de categoria. Os valores válidos são: - *BaselineCategories* - *DriverCategories* - *AppCategories* - *GlobalCondition* - *CatalogCategories*
-*Name<String[]>*	Especifica um nome para a categoria de configuração.
<*CommonParameters*>	Este *cmdlet* suporta os parâmetros comuns: -*Verbose*, -*Debug*, -*ErrorAction*, -*ErrorVariable*, -*OutBuffer*, -*OutVariable*.

3.1.4.1.6 Import-CMDriver

Para importar os *Drivers* para a *Task Sequence*. Importa *Drivers* para o *Driver catalog*. Como parte do processo de importação do *Driver*, o SCCM lê o *provider*, classe, versão, assinatura, *Hardware* suportado, e informação da plataforma suportada que está associada ao dispositivo. Por defeito, o *Driver* é nomeado depois do primeiro dispositivo de *Hardware* que suporta; contudo, podemos renomeá-lo depois. A lista de plataformas suportadas é baseada na informação do ficheiro INF do *Driver*. Quando importamos *Drivers* para o *catalog*, podemos adicioná-los a *Driver Packages* ou *boot image Packages*.

Sintaxe [32]:

```
Parameter Set: New Driver

Import-CM Driver -UncFileLocation <String> [-AdministrativeCategory
<IResult Object[]> ] [-BootImagePackage <IResult Object[]> ] [- Driver Package
<IResult Object[]> ] [-EnableAndAllowInstall <Boolean> ] [-
ImportDuplicate Driver Option <ImportDuplicate Driver Option> {AppendCategory |
KeepExistingCategory | NotImport | OverwriteCategory} ] [-
SupportedPlatformName <String[]> ] [-
Update DistributionPointsforBootImagePackage <Boolean> ] [-
Update DistributionPointsfor Driver Package <Boolean> ] [-Confirm] [-WhatIf] [
<CommonParameters>]
```

Tabela 12 - Parâmetros usados no *cmdlet* Import-CM*Driver*

Parâmetros usados	**Descrição**
-AdministrativeCategory <IResultObject[]>	Especifica um *array* de categorias administrativas. Atribuir os *Drivers* a uma categoria administrative para efeitos de filtragem, como por exemplo as categorias "*Desktops*" ou "Notebooks". Para obter o objeto da categoria administrativa category *Object*, podemos usar o *cmdlet* Get-CMCategory.
-DriverPackage<IResultObject[]>	Especifica um *array* de objetos *Driver package*. Este parâmetro usa-se para especificar os *Driver Packages* que o *Configuration Manager* usa para distribuir os *Drivers*. Para obter o objeto *Driver package*, podemos usar o *cmdlet* Get-CM*Driver*Package.
-EnableAndAllowInstall<Boolean>	Indica se o *Configuration Manager* disponibiliza os *Drivers* e permite aos computadores instalá-los.
-ImportDuplicateDriverOption <ImportDuplicateDriverOption>	Especifica como o *Configuration Manager* gere as categorias de *Drivers* quando importamos um *Driver* duplicado. Os valores válidos são: - *AppendCategory* - *KeepExistingCategory*

	- *NotImport* - *OverwriteCategory*
-*UncFileLocation<String>*	Especifica o caminho de rede (UNC) dos *Drivers*. Para importar todos os *Drivers* contidos numa determinada pasta, especifica-se o caminho de rede para a pasta. Por exemplo: \\servidor\pasta. Para importar um *Driver* específico de uma pasta, especifica-se o caminho de rede (UNC) para o ficheiro .INF do *Driver* ou para o Txtsetup.oem.
<*CommonParameters*>	Este *cmdlet* suporta os parâmetros comuns: -*Verbose*, -*Debug*, -*ErrorAction*, -*ErrorVariable*, -*OutBuffer*, -*OutVariable*.

3.1.5 *Command shell*

A *command shell* [33] ou *Windows command line* é uma aplicação que disponibiliza comunicação direta entre utilizador e Sistema Operativo. É uma interface não gráfica que fornece um ambiente no qual corremos as aplicações e utilitários com base em caracteres. A *command shell* executa os programas e apresenta o seu *output* no ecrã, através do uso de caracteres individuais, de forma semelhante ao interpretador de comandos do MS-DOS ("MS-DOS *command interpreter*"), *Command*.com. A *command shell* no Sistema Operativo *Windows* usa o interpretador de comandos Cmd.exe. O Cmd.exe carrega aplicações, direciona o fluxo de informação entre aplicações e traduz o *input* do utilizador para uma forma que o Sistema Operativo compreenda.

Podemos usar a *command shell* para criar e editar *Scripts* para automatizar tarefas rotineiras. Por exemplo, podemos criar *Scripts* em ficheiros *batch* (.bat) para automatizar a gestão de contas de utilizador ou *backups* noturnos. Também podemos usar a versão *command-line* do *Windows Script Host* [34] para correr *Scripts* mais sofisticados na *command shell*.

Podemos utilizar o *Windows Script Host* para executar *Scripts* clicando num ficheiro de *Script* no ambiente de trabalho do *Windows* ou escrevendo o nome de um ficheiro de *Script* na linha de comandos. O *Windows Script Host* tem poucos requisitos de memória e é ideal para o processamento de *Scripts* interativos e não interativos, tais como os *Scripts* de início de sessão e os *Scripts* de administração.

Existem duas versões do *Windows Script Host*: uma versão baseada no *Windows* (Wscript.exe), que fornece uma folha de propriedades para definir as propriedades de *Scripts* [35], e uma versão baseada na linha de comandos (Cscript.exe), que fornece

parâmetros da linha de comandos para definir as propriedades de *Scripts* [36]. Podemos executar qualquer uma destas versões escrevendo Wscript.exe ou Cscript.exe na linha de comandos.

Anteriormente, a única linguagem de *Scripts* nativa suportada pelo sistema operativo *Windows* era a dos ficheiros *batch*. Embora a linguagem *batch* seja rápida e ocupe menos espaço, as suas funcionalidades são limitadas quando comparadas com as linguagens *VBScript*, *JScript* e mais recentemente *Powershell*. Por exemplo, a capacidade de controlar o fluxo de um programa não estava incorporada na linguagem *batch*. Actualmente a arquitectura do *Windows Script Host* permite que os utilizadores tirem partido destas poderosas linguagens de *Scripts*, apesar de continuar a ser fornecido suporte para ficheiros *batch*.

Podemos realizar as operações mais eficientemente usando *Scripts* do que uma interface de utilizador. Os *Scripts* aceitam todos os comandos disponíveis na *command line* [37].

3.1.6 *Console Registry Tool for Windows (REG.EXE)*

O Registo do *Windows* (“*Windows Registry”)* é uma base de dados hierárquica central utilizada no *Windows* para guardar informações necessárias para configurar o sistema para um ou mais utilizadores, aplicações e dispositivos de *Hardware*. Contém informações que o *Windows* referencia continuamente durante o funcionamento, tais como os perfis de cada utilizador, as aplicações instaladas no computador e os tipos de documentos que cada uma pode criar, definições de folhas de propriedades para pastas e ícones de aplicações, o *Hardware* existente no sistema e as portas que estão a ser utilizadas. Tem uma organização parecida com uma pasta no computador, com as suas subpastas (neste caso as "*root keys*" e documentos (neste caso as subchaves) dentro destas [38].

A *Microsoft* tem uma ferramenta, chamada "*Console Registry Tool for Windows* (REG.EXE)" que permite pesquisar, incluir, excluir e até guardar chaves do Registo do *Windows* através de linha de comando, o que nos permite adaptá-la a um arquivo *batch* (.BAT) ou a uma rotina de *Script* [39][40].

Com o comando Reg add podemos adicionar novas subchaves e entradas ao registo, da seguinte forma [41]:

```
REG ADD [\\Machine\]KeyName [/v ValueName | /ve] [/t Type] [/s Separator] [/d
Data] [/f]
```

As entradas de registo para as opções de configuração das atualizações automáticas estão localizadas na seguinte subchave [42]:

```
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate\AU
```

As chaves e os valores úteis para a solução desenvolvida estão listadas na tabela seguinte:

Nome da entrada	**Tipo**	**Valores**
AUOptions	Reg_DWORD	Range = 2|3|4|5 2 = Notificar antes do *Download*. 3 = *Download* automático e notificação de instalação. 4 = *Download* automático e agendamento de instalação. Válido apenas se existirem valores para *ScheduledInstallDay* e *ScheduledInstallTime*. 5 = As atualizações automáticas são necessárias e os utilizadores podem configurá-las.

3.1.7 *Windows PreInstallation Environment (Windows PE)*

O *Windows PreInstallation Environment* (*Windows* PE), em português: "Ambiente de Pré-instalação do *Windows*", é um Sistema Operativo mínimo, desenvolvido para preparar um computador que não possui um Sistema Operativo instalado. Com o *Windows* PE podemos particionar e formatar os discos rígidos, aplicar imagens do *Windows* ao computador local, executar ferramentas de diagnóstico, solucionar problemas, e também iniciar a Instalação do *Windows* localmente ou através de um compartilhamento de rede.

O *Windows* PE suporta um subconjunto das API e serviços do *Windows*. Não é projetado para ser o Sistema Operativo principal, sendo usado como um ambiente de pré-instalação autónomo e como tecnologia base de outras tecnologias de instalação e recuperação, como por exemplo, o WDS (*Windows Deployment Services*) e WinRE (Ambiente de Recuperação do *Windows*).

Requisitos e considerações sobre o *Windows* PE:

- O *Windows* PE é executado diretamente da memória RAM. Um disco virtual com a letra de unidade X é criado para o *Windows* PE, sendo necessários, no mínimo, 512 MB de RAM;
- É possível gravar uma imagem do *Windows* PE num CD/DVD ou unidade *flash* USB (*pen drive*). Caso se use uma unidade *flash*, o sistema de ficheiros deverá ser FAT32 e a partição deve ser marcada como ativa;
- O tempo máximo de execução de uma sessão é de 72 horas;
- Suporte nativo para o sistema de ficheiros NTFS 5.x, incluindo a criação e a gestão de volume dinâmico;

- Suporte nativo a TCP/IP e compartilhamento de ficheiros (somente cliente);
- Suporte nativo para *Drivers* de dispositivo do *Windows* (32 bits ou 64 bits);
- Suporte nativo a um subconjunto de API do *Windows*;
- Suporte opcional para WMI, MDAC (*Windows Data Access Components*), HTAs (aplicativos HTML), *Powershell*, e .NET *Framework*;
- A capacidade de iniciar a partir de vários tipos de media, incluindo CD, DVD, unidade *flash* USB, e um Servidor de *Deployment* do *Windows*;
- Suporte para manutenção de imagens *offline*;
- Instalação *offline* de imagens;
- Inclusão de todos os *Drivers* do *Hyper-V*, exceto *Drivers* de vídeo. Recursos com suporte incluem armazenamento em massa, integração do rato, e adaptadores de rede [43].

3.1.8 *Deployment Image Servicing and Management (DISM)*

Deployment Image Servicing and Management (DISM) é uma ferramenta de linha de comandos utilizada para montar e atualizar imagens do *Windows* antes da implementação. Podemos utilizar os comandos de gestão de imagens do DISM para montar e obter informações sobre ficheiros de imagem do *Windows* (.wim) ou discos rígidos virtuais (VHD), e para capturar, dividir ou, sintetizando, gerir ficheiros .wim.

Também podemos utilizar o DISM para instalar, desinstalar, configurar e atualizar funcionalidades, pacotes, controladores e definições internacionais do *Windows* num ficheiro .wim ou VHD com os comandos de atualização do DISM.

Os comandos do DISM são utilizados em imagens *offline*, mas também há subconjuntos de comandos do DISM disponíveis para levar a cabo a atualização de um sistema operativo em execução [44].

Exemplo de funcionamento do DISM para Criar uma imagem personalizada do *Windows* PE [43]:

A primeira etapa para criar uma imagem personalizada do *Windows* PE é modificar a imagem base do *Windows* PE usando a ferramenta DISM, que extrai os *ficheiros* para um diretório local e permite adicionar e remover componentes opcionais e pacotes de idiomas. Permite também adicionar *Drivers* de terceiros.

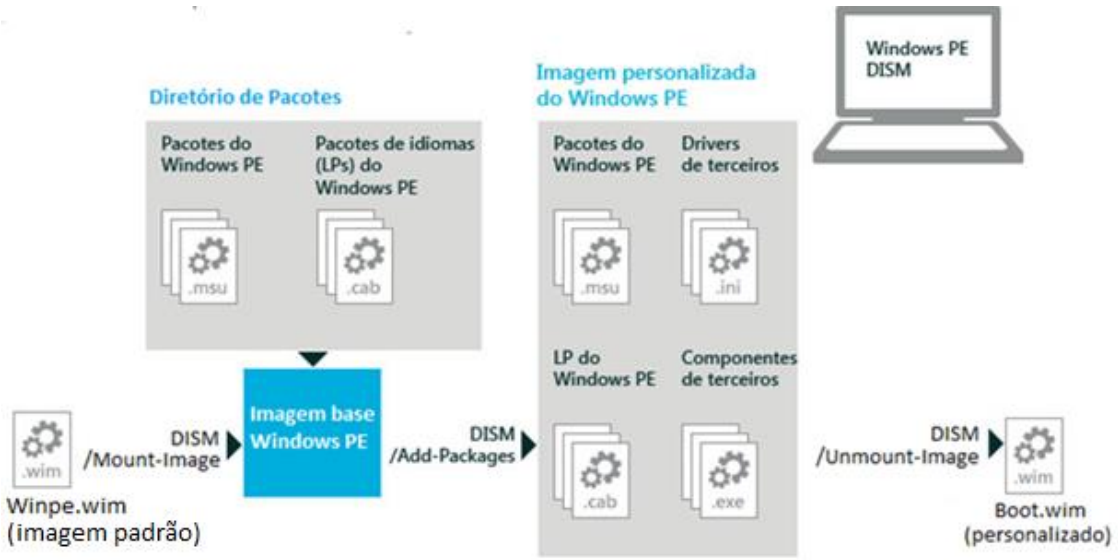


Figura 11 - Visão geral do processo de criação de uma imagem personalizada do *Windows* PE, recorrendo ao DISM

3.1.9 *Ferramentas de gestão de Drivers dos respetivos fabricantes*

3.1.9.1 HP *SoftPaq Download Manager* (SDM)

É uma ferramenta que disponibiliza uma forma simples e poderosa de obter as atualizações para os modelos HP. Pode reduzir significativamente o tempo necessário para encontrar e obter as atualizações, pois permite evitar o processo manual de *Download*, extração e instalação, tornando possível obter os *SoftPaqs*[15] apropriados para cada um dos modelos, de uma vez só, em vez de procurar cada um deles individualmente [45].

3.1.9.2 Lenovo *Update Retriever*

É uma ferramenta semelhante à anterior, que possibilita obter os *Packages* das atualizações do "Centro de ajuda" na página *Web* da Lenovo, para um repositório, numa pasta partilhada na rede, o que permite maior controlo das atualizações disponíveis. A

15 Nome que a HP atribui aos seus ficheiros que contêm os *drivers* e respetivo *software* para gestão dos mesmos.

pesquisa pelos *Packages* das atualizações é um processo que pode ser agendado ou iniciado manualmente. O *Update Retriever* pode ser configurado para apresentar uma notificação quando encontrar novas atualizações [46].

3.1.9.3 HP *System Software Manager* (SSM)

É um utilitário que deteta e atualiza automaticamente a BIOS, *Drivers* de dispositivos e versões de agentes de gestão nos computadores da rede. Foi desenvolvido para simplificar o *Deployment* dos *SoftPaqs* da HP para os computadores. Deteta qual dos *SoftPaqs* disponíveis são aplicáveis a cada Sistema e instala-os. O HP SSM acede aos *SoftPaqs* da rede local ou pasta local, em vez da internet, reduzindo o tempo de actualização e tráfego [45].

3.1.9.4 Lenovo *Thin Installer*

O *Thin Installer* foi desenvolvido para ajudar os administradores a manter os sistemas atualizados. O programa procura por atualizações num repositório criado previamente. Este repositório pode ser num disco local, numa partilha de rede ("*network share*"), numa partilha *Web* ("*Web share*") (HTTP), ou CD, DVD, *drive* USB. O *Thin Installer* reconhece automaticamente o tipo de máquina, número de modelo e restante informação do sistema para determinar se o mesmo necessita das atualizações que estiverem disponíveis. Se duas ou mais atualizações estiverem relacionadas com o modelo, o programa obtém e instala as mesmas. Pode ser configurado para instalar as atualizações manual ou automaticamente. Não requer instalação e não cria quaisquer chaves no registo [47].

O funcionamento destas ferramentas é idêntico para ambos os fabricantes:

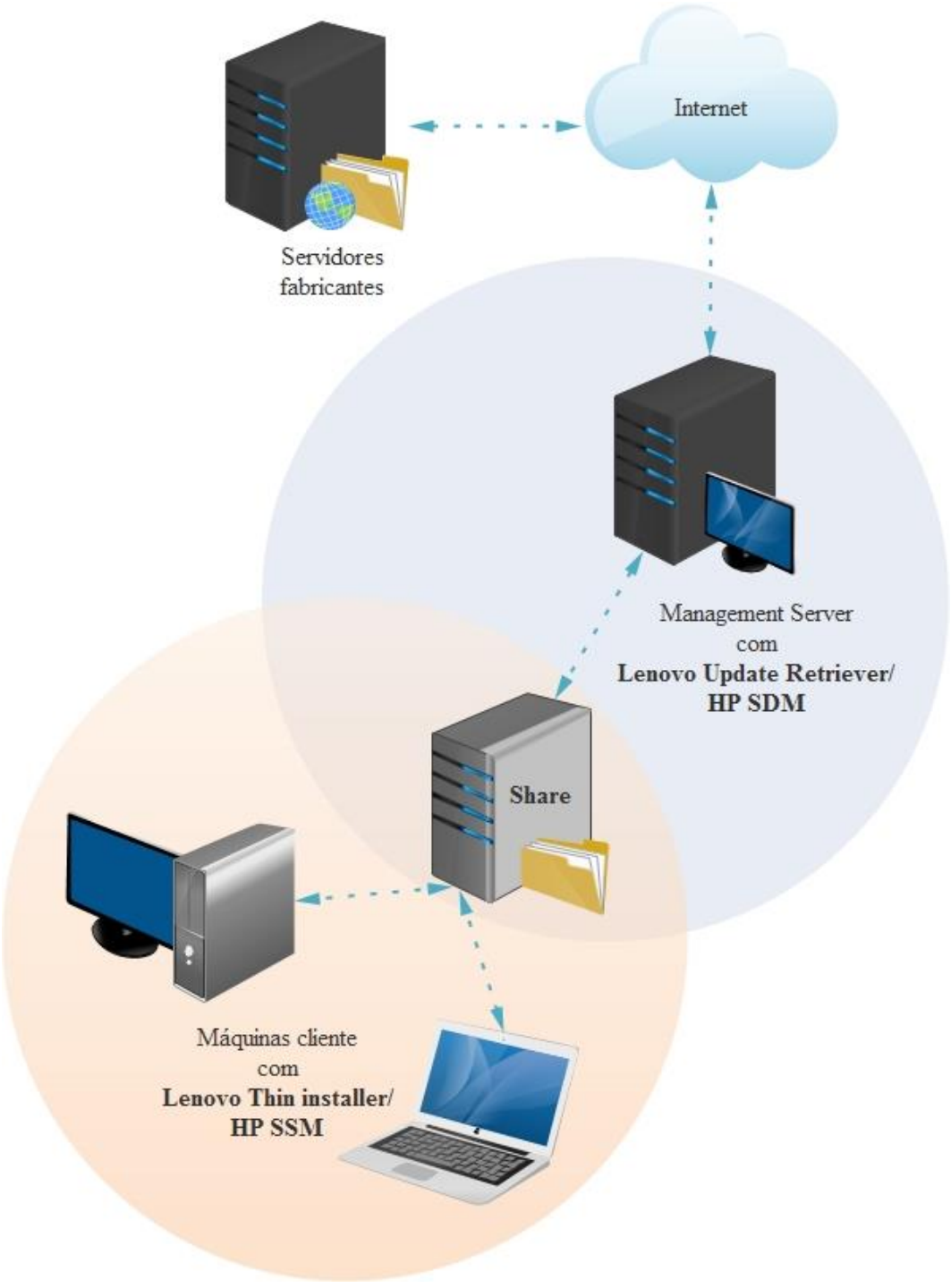


Figura 12 - Arquitetura de funcionamento das ferramentas dos fabricantes no processo de instalação de *Drivers*.

3.1.10 *Double Driver*

O *Double Driver* é uma ferramenta muito simples que, não só permite ver todos os *Drivers* instalados no Sistema, mas também capturar, restaurar, guardar e imprimir todos os *Drivers* escolhidos.

O *Double Driver* analisa o Sistema e lista os detalhes mais importantes dos *Drivers*, como versão, *data*, fornecedor, etc. *To*dos os *Drivers* encontrados podem ser facilmente capturados e restaurados [48].

Lista de funcionalidades do *Double Driver*:

- Lista, guarda, e imprime detalhes dos *Drivers*;
- Captura *Drivers* do Sistema corrente (em execução);
- Captura *Drivers* de Sistemas *non-live/non-booting*;
- Captura *Drivers* para pastas estruturadas, pastas comprimidas e *self-extracting*;
- Restaura *Drivers* de capturas anteriores;
- Disponível em GUI e CLI;
- *Portable* (não é necessária instalação).

Utilização:

```
ddc <comando> [<swicthes>]
```

Comandos:

b - *Backup Drivers*

r - *Restore Drivers*

parâmetros:

/*source*:[*path*] :: *Source Directory* (*drive*:*path* or *Server**share**path*)

/*target*:[*path*] :: *Target Directory* (*drive*:*path* or *Server**share**path*)

3.1.11 PnPUtil

A ferramenta PnPUtil (PnPUtil.exe) é uma ferramenta de linha de commando que permite a um administrador executar as seguintes ações em *Driver* packages:

- Adicionar um *Driver package* à *Driver* store;
- Eliminar um *Driver package* da *Driver* store;
- Enumerar os *Driver* packages que estão na *Driver* store. Apenas *Driver* packages que não são *in-box* packages são listados. Um *in-box Driver package* é

um *Driver package* que está inclido na instalação por defeito do *Windows* ou dos seus *Service* Packs.

3.1.12 *Microsoft Signtool*

Uma abordagem para fornecer garantias de autenticidade e integridade a um ficheiro é anexar-lhe uma assinatura digital. Uma assinatura digital anexada a um ficheiro identifica positivamente o distribuidor desse ficheiro e garante que os conteúdos não foram alterados após a assinatura ser criada [49].

A ferramenta *Signtool* da *Microsoft* é uma ferramenta de linha de comandos que permite:

- Assinar ficheiros digitalmente;
- Verificar a assinatura de ficheiros;
- Imprimir uma data e uma hora, para validação cronológica de ficheiros [50].

A sintaxe da *Signtool* é a seguinte:

```
Signtool [Command][Options][FileName …]
```

Os seguintes comandos foram usados:

Comando	Descrição
verify	Verifica a assinatura de ficheiros

O comando "*verify*", determina se o certificado de assinatura foi emitido por uma entidade confiável, se foi revogado e, opcionalmente, se é válido para uma política específica.

A *Signtool* retorna um código de saída ("*exit code*"):

- *Zero*, em caso de execução bem sucedida;
- um, em caso de execução falhada;
- dois, se a execução terminar com avisos.

Contudo, se a *Signtool* encontrar uma exceção sem tratamento ("*unhandled Exception*"), o valor retornado é indefinido ("*Undefined*").

Temos ainda as seguintes opções úteis, a nível de visualização (que se aplicam a todos os comandos do *Signtool*):

Tabela 13 - Opções de visualização aplicáveis a todos os comandos do *Signtool*

Opção	Descrição
/*Debug*	Apresenta informação de *Debugging*[16].
/q	Não apresenta *output* em caso de execução bem sucedida e apresenta um *output* mínimo em caso de execução falhada.
/v	Apresenta *output* detalhado em caso de execução bem sucedida, execução falhada e mensagens de aviso.

3.1.13 *Microsoft Driver Verifier*

Driver Verifier é uma ferramenta disponibilizada pela *Microsoft*, que vem incluída nos Sistemas Operativos *Windows*, desde a versão *Windows* 2000.

O *Driver Verifier* monitoriza os *Drivers* em *Windows Kernel-mode* para detetar chamadas ilegais a funções ("*illegal Function calls*") ou ações que possam corromper o Sistema. Permite sujeitar os *Drivers* a uma variedade de testes, para detetar comportamentos impróprios.

Pode verificar vários *Drivers* simultaneamente, ou um *Driver* de cada vez.

Podemos configurar que testes executar, o que permite submeter os *Drivers* a testes de *stress* pesados ou a testes mais simples [51].

A localização do executável é: %windir%*System*32.

Deverá ser executado via linha de comando, em modo Administrador.

Sintaxe [52]:

```
Verifier /standard /Driver <name>[<name>...]Verifier /standard /allverifier
[/flags <flags>] [/ [<probability> de
falhas[<tags>[<applications>[<minutes>]] /Driver <name>[<name>...]Verifier
[/flags FLAGS] [/ [<probability> de falhas[<tags>[<applications>[<minutes>]]
/allverifier /Querysettingsverifier /volatile /flags <flags> Verifier
/volatile /addDriver <name>[<name>...]Verifier /volatile /removeDriver
<name>[<name>...]Verifier /volatile / [<probability> de
falhas[<tags>[<applications>]/Queryverifier de /resetverifier de verificador
/log <LogFileName>[/ intervalo <seconds>]
```

Utilização:

```
Verifier [/volatile] [/standard | /flags Options] [/all | /Driver DriverList]
```

Opções:

---

[16] Em português: depuração, é um processo metódico de localizar e corrigir erros.

/*standard* - opções por defeito (que serão ativadas após reiniciar):

• *Special Pool*;

• *Force* IRQL *checking*;

• *Pool tracking*;

• I/O *verification*;

• *Deadlock detection*;

• DMA *verification*;

• *Security checks*;

• *Miscellaneous checks*.

/*all* - verifica todos os *Drivers* instalados (após reiniciar)

/faults - ativa a simulação de baixos recursos ("*Low resources simulation*")

Para verificar apenas alguns *Drivers* (não aceita *wildcards*; são necessários os nomes completos):

```
/Driver DriverList
```

*Driver*List - lista de *Drivers* em nome binário, como *Driver*.sys, separados por espaço.

Para excluir alguns; caso estejam todos selecionados com /*all*:

```
/Driver.exclude DriverList
```

Eliminar todas as configurações (após reiniciar, já não irá testar os *Drivers*):

```
Verifier /reset (forçar restart após este comando)
```

O *DUMP File* gerado pelo *Driver Verifier* (em caso de problemas) estará em:

%*SystemRoot*%\MEMORY.DMP

Se for um *Minidump*:

%*SystemRoot*%*Minidump*\Mini000000-01.dmp

Tabela 14 - Tipos de *DUMP File* possíveis de obter do *Driver Verifier* (em caso de problemas)

Nome	Descrição
Small memory *DUMP*	Menor tamanho. Informação limitada. Erros que não estejam diretamente causados pela *thread* que estava a correr quando se deu o problema podem não ser descobertos na análise deste ficheiro.

Kernel memory *DUMP*	Este ficheiro *DUMP* não inclui memória não alocada ou memória alocada às aplicações instaladas. Inclui apenas memória alocada ao *kernel* e *Hardware Abstraction Layer* (HAL). Se ocorrer um problema com uma aplicação (e não SO) este tipo de *DUMP* não é útil.
Complete memory *DUMP*	Um *complete memory DUMP* guarda todo o conteúdo da memória de um Sistema quando o computador pára inesperadamente. Este tipo de *DUMP* permite uma análise completa sobre o que causou o *crash.*

Podemos também ativar a criação de *logs*, através do *Switch*:

/logging - Ativa os relatórios de regras violadas, detetadas pelas extensões selecionadas.

3.1.14 *Microsoft Windows* Debugger

O *Windows* Debugger (*Windbg*) é um debugger para *Windows*, distribuído pela *Microsoft*. Pode ser usado para fazer *debug* de aplicações e *Drivers* em "*user mode*" e do próprio Sistema Operativo em "*kernel mode*".

Permite analisar dumps de memória ou sistema, criados por crashes do sistema. Podem ser introduzidos comandos para revelar certos aspectos do estado do Sistema quando ocorrer o *crash*.

Está incluído no Sistema Operativo *Microsoft Windows*, desde a versão 2000.

É usado pela equipa *Microsoft Windows* para desenvolver o próprio *Windows*.

O seu motor de *debug* faz parte do *Windows*.

Sintaxe [53]:

```
Windbg [ -server ServerTransport | -remote ClientTransport ] [-lsrcpath ]
   [ -premote SmartClientTransport ] [-?] [-ee {masm|c++}]
   [-clines lines] [-b] [-d] [-aExtension] [-e Event]
   [-failinc] [-g] [-G] [-hd] [-j] [-n] [-noshell] [-o]
   [-Q | -QY] [-QS | -QSY] [-robp] [-secure] [-ses] [-sdce]
   [-sicv] [-sins] [-snc] [-snul] [-sup] [-sflags 0xNumber]
   [-T Title] [-v] [-log{o|a} LogFile] [-noinh]
   [-i ImagePath] [-y SymbolPath] [-srcpath SourcePath]
   [-k [ConnectType] | -kl | -kx ExdiOptions] [-c "command"]
   [-pb] [-pd] [-pe] [-pr] [-pt Seconds] [-pv]
   [-W Workspace] [-WF Filename] [-WX] [-zp PageFile]
   [ -p PID | -pn Name | -psn ServiceName | -z DumpFile | executable ]
```

Parâmetro usado	**Descrição**
-z DumpFile	Especifica o nome do ficheiro de *crash DUMP* a depurar. Se o caminho ou o nome do ficheiro contiver espaços, deve estar envolvido em aspas. É possível abrir vários ficheiros *DUMP* de uma só vez, se incluirmos múltiplas opcções -z options, cada uma seguida de um nome *DumpFile.*

3.2 Trabalho realizado

3.2.1 *Análise do problema*

Um cenário de *Deployment* envolve normalmente a entrega dos seguintes elementos:

- Sistema Operativo;
- *Drivers*;
- Aplicações;
- *Language packs*;
- *Patches* para o Sistema Operativo e para as aplicações;
- Customizações do Sistema Operativo e aplicações.

A entrega de *Drivers* é uma das tarefas mais desafiantes dentro dos cenários de *Deployment*. Uma das razões é o facto de as organizações com um número significativo de computadores, os adquirirem ao longo do tempo, o que resulta numa panóplia de diferentes modelos e configurações, de vários fabricantes, podendo até incluir sistemas personalizados e sem marca específica. Consequentemente, vários *Drivers* serão necessários para suportar todos estes diferentes sistemas.

Outra razão para os *Drivers* adicionarem complexidade aos *Deployment*s é porque podem não ser necessários apenas para o funcionamento do Sistema Operativo instalado, mas antes mesmo, para ser possível fazer iniciar o computador e permitir efetuar a instalação desse mesmo Sistema Operativo (geralmente *Drivers* de rede).

Mais uma razão é que podem por vezes surgir incompatibilidades. Por exemplo, há situações em que:

- Instalar o *Driver* errado num sistema pode originar um erro e consequentemente um "*Blue screen*";
- Instalar dois *Drivers* similares (um recente e outro mais antigo) para o mesmo *Hardware* pode fazer com que seja instalado o *Driver* errado, porque o fabricante concebeu um *Driver* inadequadamente;
- O fabricante lança um novo *Driver* para uma versão de *Hardware* mais atual e diz que o novo *Driver* suporta a versão de *Hardware* anterior, mas quando testamos verificarmos que o *Hardware* mais antigo não funciona corretamente com o novo *Driver*.

Por outro lado, como alguns fabricantes gostam de integrar aplicações de gestão de *Drivers* com os mesmos, o que implica terem de ser instalados no final do processo de *Deployment*, executando estas aplicações, o que exige trabalho adicional com a *Task*

Sequence, uma vez obtido o *Driver*, é necessário extrair os ficheiros do mesmo por forma a ser possível importá-los para o *Deployment share*, porque no SCCM apenas conseguimos importar *Drivers* através dos ficheiros *".inf"* ou *".oem"*.

Finalmente, quando se faz *deploy* da última versão do *Windows*, tanto com *Hardware* muito recente, como muito antigo, podemos frequentemente ter problemas com *Drivers*. Por exemplo, no caso do *Windows* 7, que é o Sistema Operativo atualmente usado no âmbito do projeto, apesar de incluir milhares de *Drivers*, pode não ter os adequados para o *Hardware* comercializado depois do *Release To Manufacturing* (RTM) do *Windows* 7, como, numa situação mais genérica, pode também não incluir os *Drivers* adequados para *Hardware* da era do XP ou anterior.

Adicionalmente também pode ser difícil encontrar o *Driver* indicado no *website* do fabricante, ou o mesmo pode já nem sequer existir, no caso de *Hardware* mais antigo.

O primeiro desafio é mesmo encontrar os *Drivers Out-of-box* que os sistemas poderão necessitar. Alguns fabricantes tornam esta tarefa mais simples que outros.

Uns disponibilizam os *Drivers*, como ficheiros "*.cab*" que podem ser extraídos para uma pasta, sendo apenas necessário indicar a mesma para fazer a importação dos *Drivers* para o *Deployment share*, outros têm ferramentas para baixar os *Drivers*, como é o caso da HP e Lenovo (marcas dos modelos em uso no CLIENTE atualmente).

Alguns fabricantes fornecem os *Drivers* em ficheiros "*.exe*" em vez de "*.cab*". Nesse caso é necessário extrair o conteúdo destes para uma pasta, pois necessitamos dos ficheiros "*.inf*" e relacionados (p. ex. os "*.dll*" chamados pelo "*.inf*").

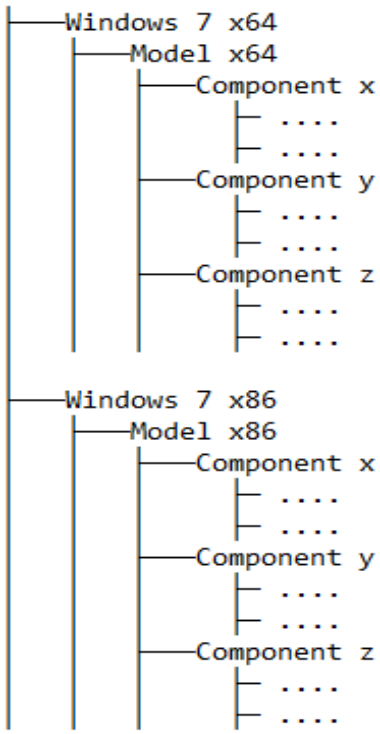
Analisado o processo manual de instalação dos *Drivers*, no âmbito da solução implementada, para os *archetypes* e respetivo Sistema Operativo em uso:

- *FAT: Windows* 7 *Enterprise (x*64*)*
- *THIN PC: Windows* 7 *Embedded (x*86*)*

Concluiu-se que a instalação dos *Drivers* para novos modelos segue o seguinte padrão:

- *Download* das versões mais recentes dos *Drivers* para o modelo e Sistema Operativo específicos;
- Instalação manual dos *Drivers* e/ou com recurso ao agente de *Drivers*, que é um *Software* do fabricante, usado no projeto para procurar/obter os *Drivers da* réplica local de *Drivers* dos fabricantes, previamente criada no servidor;
- Captura dos *Drivers* "*non-Microsoft*" através do *Software Double Driver*;
    - A captura cria uma estrutura de pastas para cada componente.
- Copiar as *sources* dos *Drivers* para a localização definida para o efeito:

- Esta localização é um Sistema de Arquivos Distribuídos, logo é necessário garantir que as várias localizações têm a mesma cópia das *sources*;
- A estrutura das *sources* é:



- Importar os *Drivers* para a *Software Library* no *SCCM:*
  - *Operating Systems\Drivers*
    1. *Import Driver*;
       a) *Locate Driver*: *Import all Drivers in the following network path (UNC)* com a opção *"Import the Driver and append a new category to the existing categories";*
       b) Detalhes do *Driver*: Activar "*Enable these Drivers and allow computers to install them"* e criar uma nova categoria com a convenção de nomes Sistema Operativo e modelo (ex: *Windows* 7 \ x64 \ Lenovo ThinkPad X1 Carbon);
       c) Adicionar o *Driver* a um *Package*: *New Package:*
          i. Nome: convenção de nomes igual ao modelo (ex: Lenovo thinkPad X1 Carbon);
          ii. *Path*: p.ex.:"\\...*sources$**Driver_Packages*\ *Windows7*\" e "x64 ou x86 \ Nome do modelo"

- Desativar "*Update Distribution Points when finished*"

iii. Skip "*Add Driver to Boot Image".*

d) Na raiz dos *Drivers*, na pasta *Windows* 7 x64 ou *Windows* 7 x86, criar uma nova pasta com nome do modelo. Mover os *Drivers* importados para a pasta;

e) Na raíz dos *Packages* de *Drivers* mover o novo *Driver package* para a pasta *Windows* 7 x64 ou *Windows* 7 x86;

f) Em *Driver package, "distribute the content to the following Distribution Points Groups":*

i. Nome do *Distribution Point/Group;*

ii. ...outros eventuais *Distribution Point/Group;*

Resultando o seguinte diagrama de atividades:

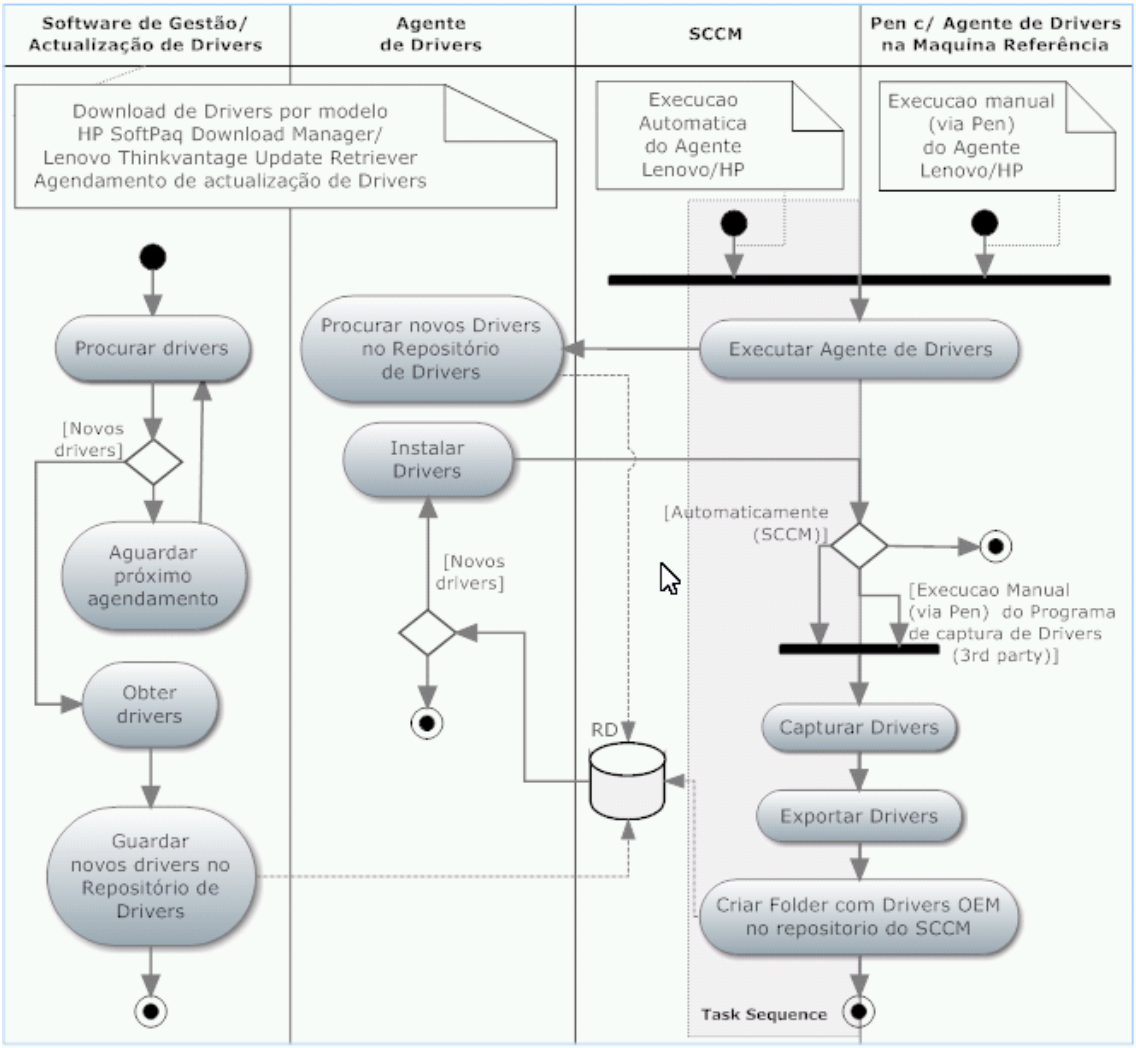


Figura 13 - Diagrama de atividades do processo manual de instalação dos Drivers

3.2.2 *Investigação e desenvolvimento da solução otimizada*

Tendo em conta o processo manual que estava a ser usado e, como base, as ferramentas já disponíveis, em uso no projeto e sabendo que o SCCM, através de uma *Task Sequence*, permite correr aplicações exteriores e também executar a *Command Line*, o que permite executar as ferramentas em modo silencioso, introduzindo os parâmetros adequados, bem como correr *Scripts*, nomeadamente de *Powershell*, desenhou-se uma possível solução para contornar ao máximo a intervenção manual neste processo.

O SCCM usa a conta local SYSTEM na instalação de software, mas também podemos usar a opção Run As para o fazer com as credenciais que especificarmos.

A conta SYSTEM ou *LocalSystem* é uma conta local pré-definida usada pelo *Service Control Manager*[17][54], que faz parte do grupo Administradores, mas é um pouco mais poderosa que a conta de Administrador e não é reconhecida pelo subsistema de segurança nem tem uma *password*.

Esta conta tem extensos privilégios no computador local e atua como o computador na rede (apresenta as credenciais do computador *Domainname\machinename*$ aos servidores remotos; isto requer *Active Directory* e dar as devidas permissões ao computador).

O seu *token* inclui os *Security Identifiers* (SIDs)[18] NT AUTHORITY\SYSTEM e BUILTIN\Administrators. Estas contas têm acesso à maioria dos objetos do sistema.

O nome desta conta em *all locales* é .*LocalSystem* ou *LocalSystem* ou *ComputerName\LocalSystem* [55].

Uma vantagem de correr um serviço na conta *LocalSystem* é que este tem acesso completo e sem restrições aos recursos locais [56].

Nas versões mais recentes do *Windows*, a maioria dos serviços do Sistema correm nesta poderosa conta.

---

[17] O *Service Control Manager* (SCM) mantém uma base de dados dos serviços instalados e serviços de drivers que permitem ao Sistema Operativo iniciar com sucesso, e disponibiliza um meio unificado e seguro de os controlar. A base de dados, que é armazenada no Registo do *Windows*, inclui informação de configuração e segurança acerca de cada serviço ou serviço de driver.

[18] O *Windows* concede ou nega acesso e privilégios a recursos baseado em *Access Control Lists* (ACLs), que usam SIDs para identificar exclusivamente utilizadores e os grupos a que pertencem. Quando um utilizador inicia a sessão num computador, é gerado um *token* de acesso que contém os SIDs do utilizador e do grupo e o nível de privilégio do utilizador. Quando um utilizador requisita acesso a um recurso, o *token* é comparado com a ACL para permitir ou negar uma ação particular num objeto particular.

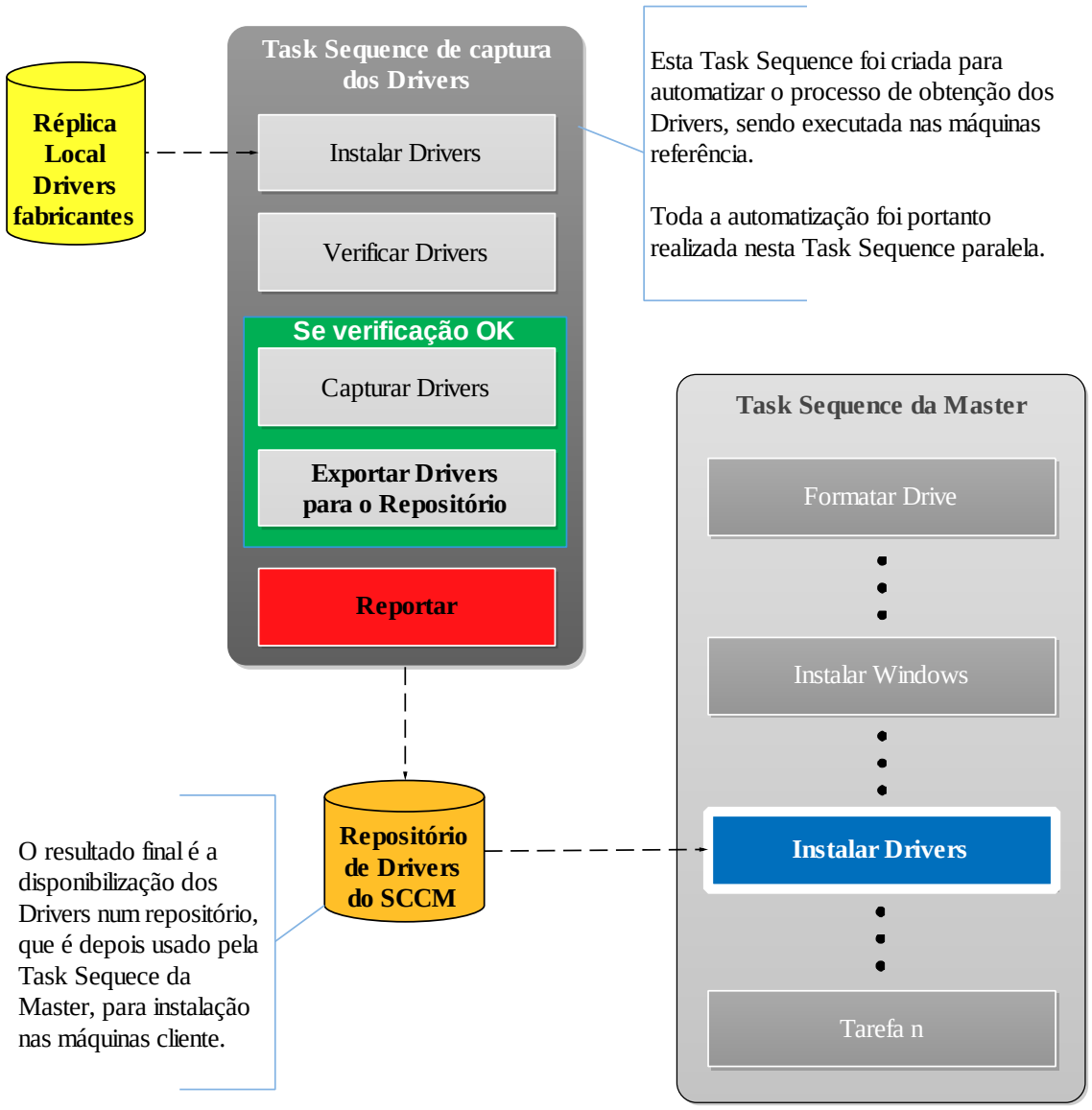


Figura 14 - Enquadramento do processo automatizado neste trabalho, na *Task Sequence* da *Master*

Mas para este processo funcionar, não bastou apenas construir uma *Task Sequence* no SCCM, pois esta constituirá apenas uma lista das tarefas necessárias, de uma forma sequencial. Foi necessário construir uma espécie de *Framework*, que funciona com recurso ao SCCM em conjugação com várias outras ferramentas, tais como:

- Ferramentas dos fabricantes dos dispositivos de *Hardware*;
- Linha de comando (*Command line*) para chamar todos os programas necessários, incluindo o *Powershell*, seja para fazer instalações de *Software*, correr *Scripts Powershell* ou introduzir alterações no registo do *Windows*;
- A linguagem de *scripting Powershell*, para programar e permitir automatizar os processos mais complexos;

- O WMI para obter as informações necessárias sobre o *Hardware*;
- Uma ferramenta de verificação de assinatura, para verificar se os *Drivers* obtidos por outras vias, que não sejam o próprio fabricante ou a *Microsoft*, estão assinados digitalmente por uma entidade confiável;
- Uma ferramenta de verificação dos *Drivers*, que realiza testes de *stress* aos *Drivers*, para verificar a possível existência de eventuais ações que possam corromper o sistema;
- Uma ferramenta de captura e backup dos *Drivers*, etc.

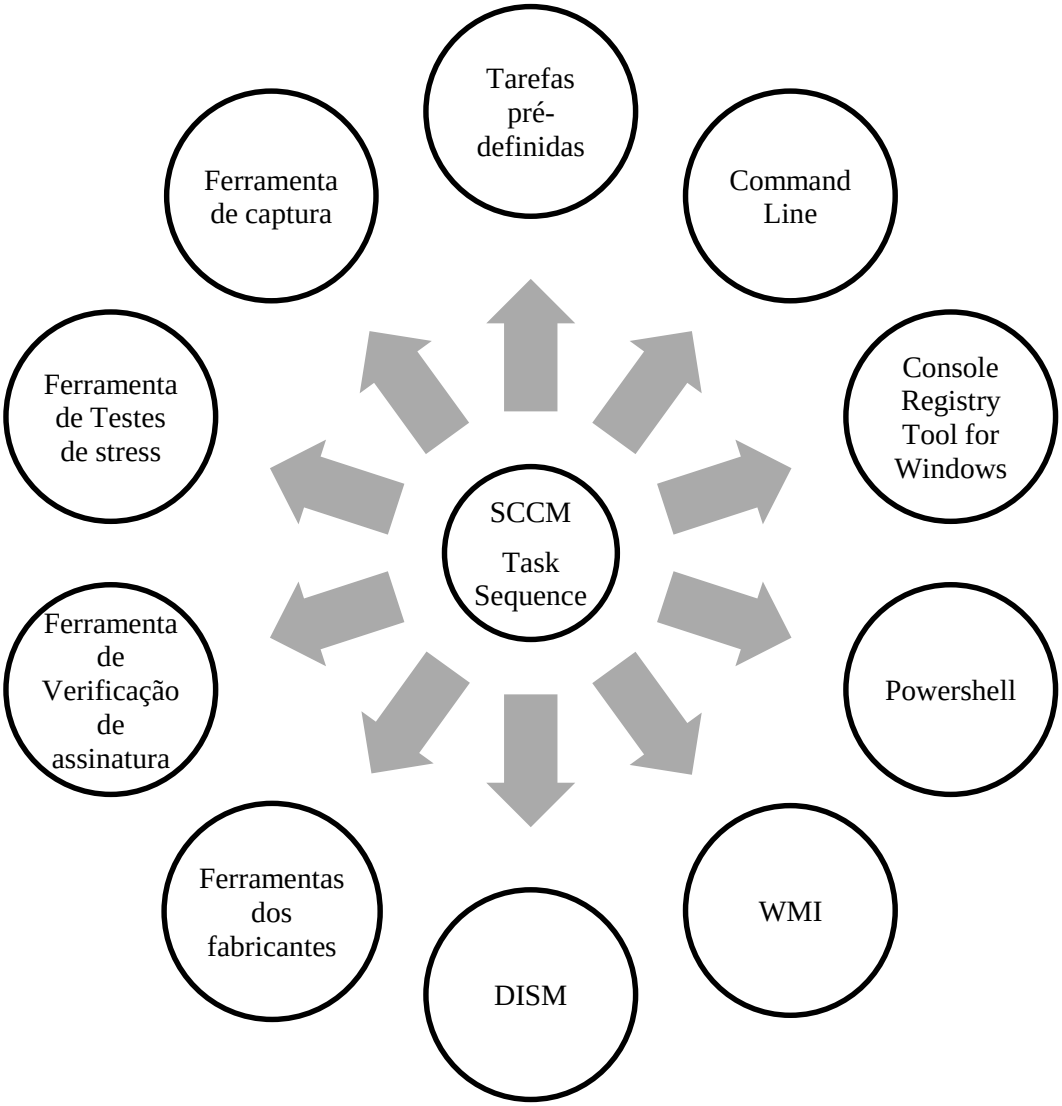


Figura 15 - O SCCM, através da *Task Sequence*, permite correr todos as restantes ferramentas utilizadas no processo, servindo de "veículo" para o mesmo.

Esta "*Framework*", na prática, consistirá numa combinação das potencialidades do SCCM, já apresentadas, com um conjunto de aplicações auxiliares previamente avaliadas, que eram utilizadas na instalação normal (manual) dos *Drivers*, integradas através de código desenvolvido de raiz, em *Powershell* e recorrendo ainda ao WMI e algumas ferramentas complementares, estas em modo automático, através dos respetivos parâmetros de automatização.

Cada tarefa criada na *Task Sequence* permite a introdução de *Scripts* de linha de comando no campo "*Command line*" e desta forma podemos manipular as várias tarefas individualmente e no seu conjunto, por forma a atingir os objetivos pretendidos, sendo desta forma possível executar todos os passos necessários, nas máquinas de referência, como:

- Instalar o Sistema;
- Executar comandos específicos e alterar configurações, como por exemplo, evitar receber os *Updates* do *Windows*, para não serem instalados *Drivers Microsoft* à partida;
- Instalar aplicações auxiliares;
- Identificar o *Hardware*;
- Instalar *Drivers*;
- Verificar o estado dos mesmos e se existem erros ou faltas;
- Aceder aos *shares* necessários e fazer backup dos *Drivers*, para a concretização do processo.

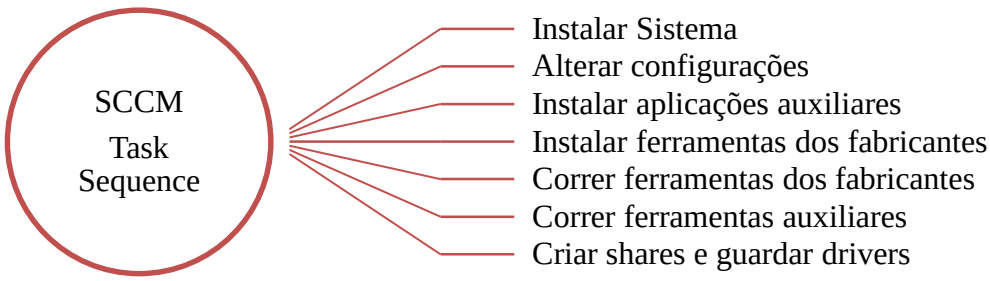


Figura 16 - O SCCM, através de uma *Task Sequence*, permite correr todas as tarefas necessárias para a concretização do processo.

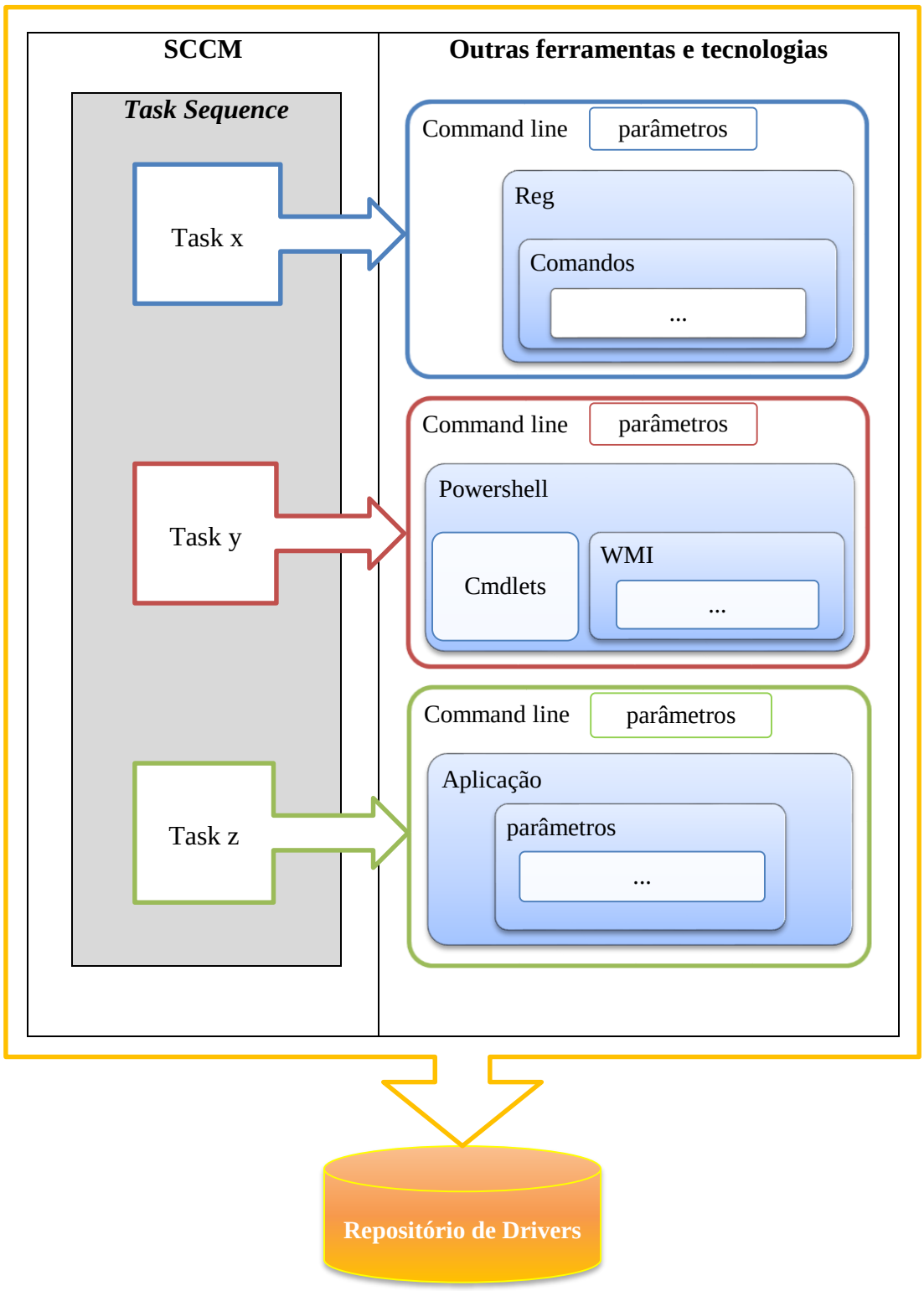


Figura 17 - Framework de automatização

O processo resultante, está detalhado no seguinte diagrama de atividades:

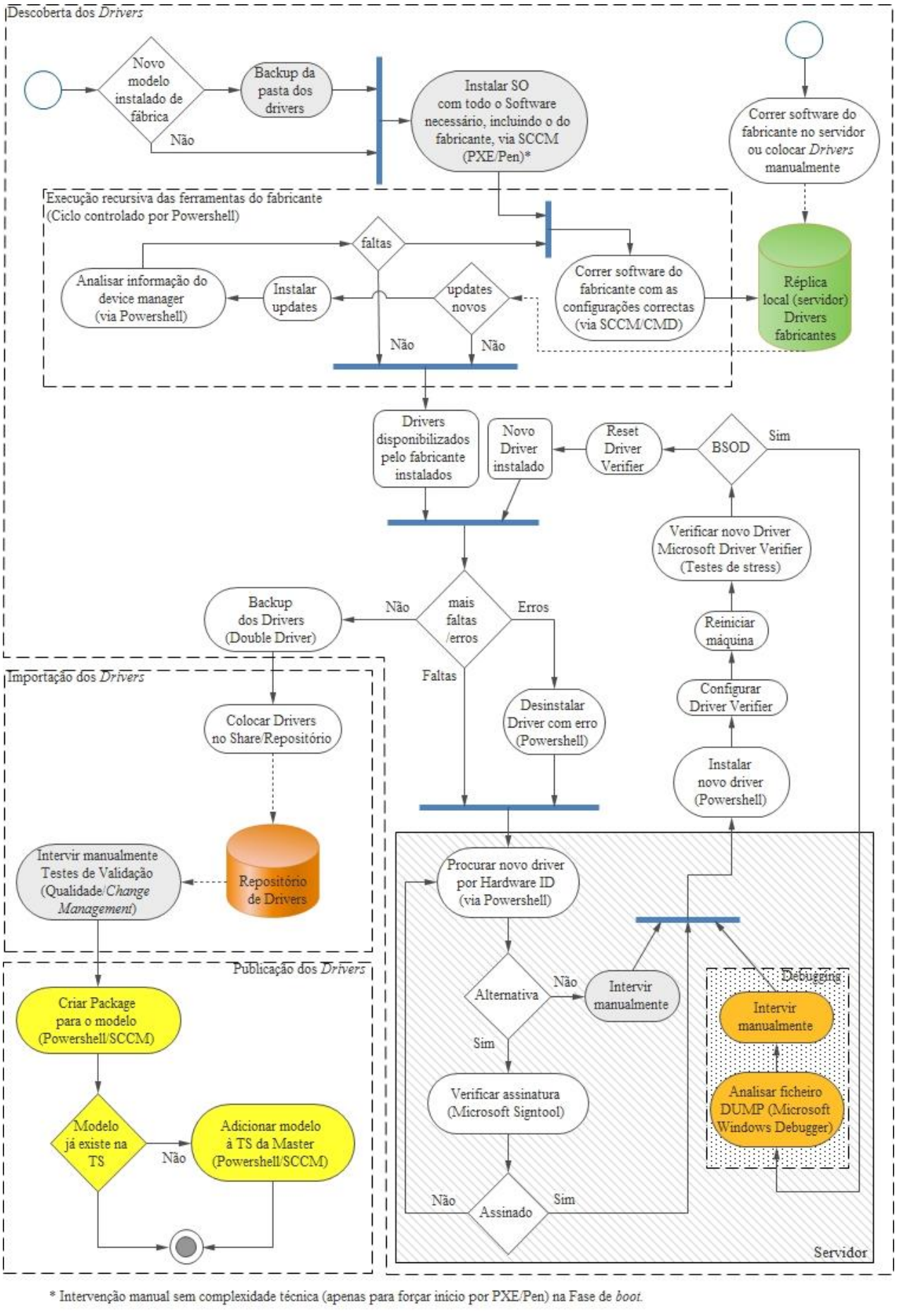


Figura 18 - Solução de semi-automatização do processo de obtenção de *Drivers* - Diagrama de atividades

3.2.3 Implementação da solução

Neste ponto é apresentado todo o processo de automatização e descritos os passos ou tarefas ("*Tasks*") chave da solução desenvolvida, resultante do meu trabalho, cuja arquitetura de funcionamente está representada na Figura seguinte:

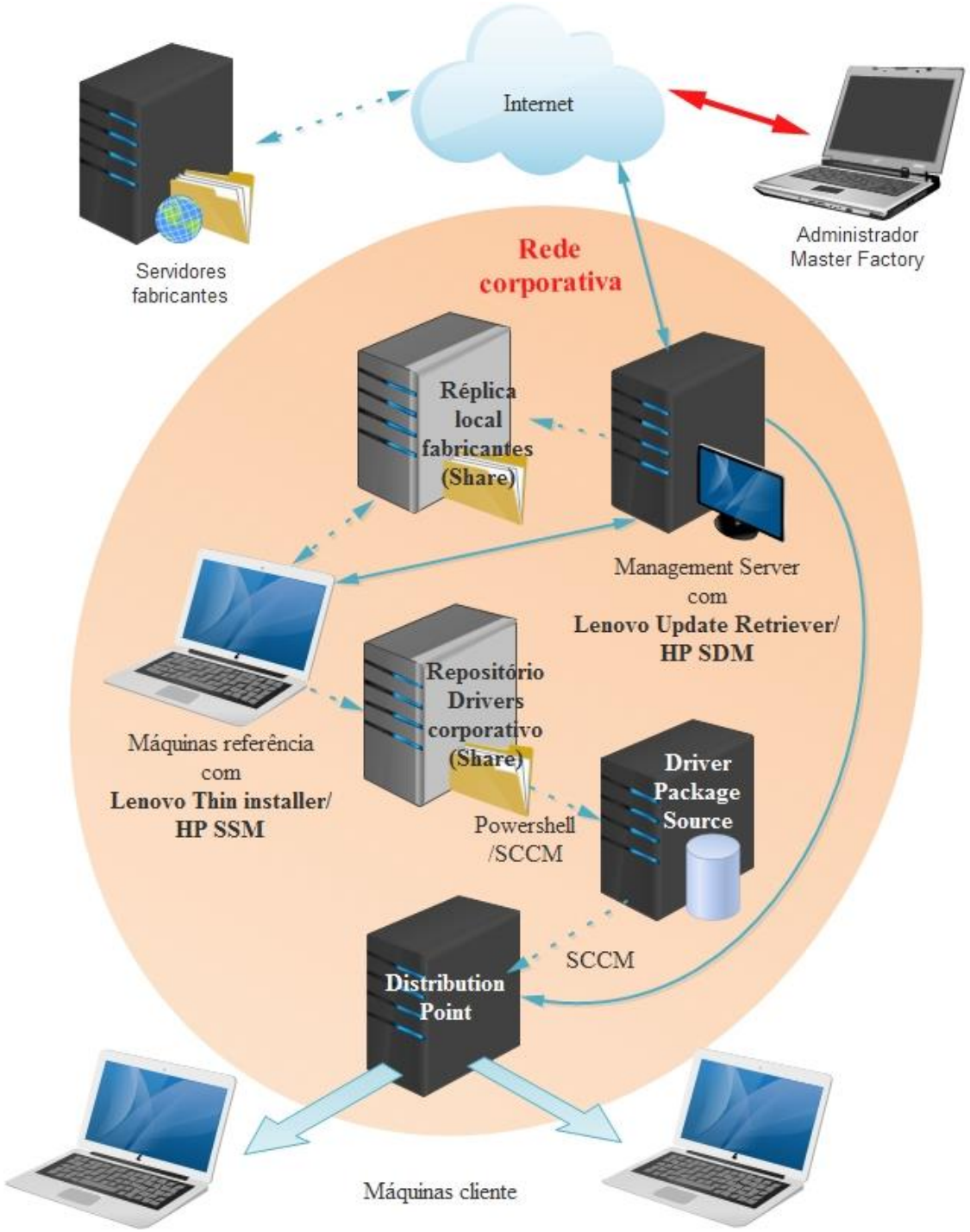


Figura 19 - Arquitetura da solução desenvolvida

Tendo em conta que o "veículo" da solução é o SCCM, através de uma *Task Sequence*, existem várias tarefas comuns que não serão descritas como, por exemplo, as que executam a formatação, instalação do Sistema Operativo e outras que, para o efeito, e apesar de essenciais, são no fundo apenas auxiliares para a solução, pelo que apenas são referenciadas.

3.2.3.1 Configuração genérica importante na *Task Sequence*

Uma vez que o principal objetivo é automatizar as operações do lado do cliente e passar a maior quantidade de informação possível para o lado do servidor, de modo a poder ser analisada e, caso necessário, haver uma intervenção corretiva de um elemento da equipa, sem necessidade de deslocação ao local, além da análise automática dos erros do Gestor de dispositivos, a informação dos *logs* que contêm informação relacionada com os procedimentos que precisamos verificar deve ser salvaguardada e passada para uma pasta partilhada ("*share*") previamente definida, no servidor.

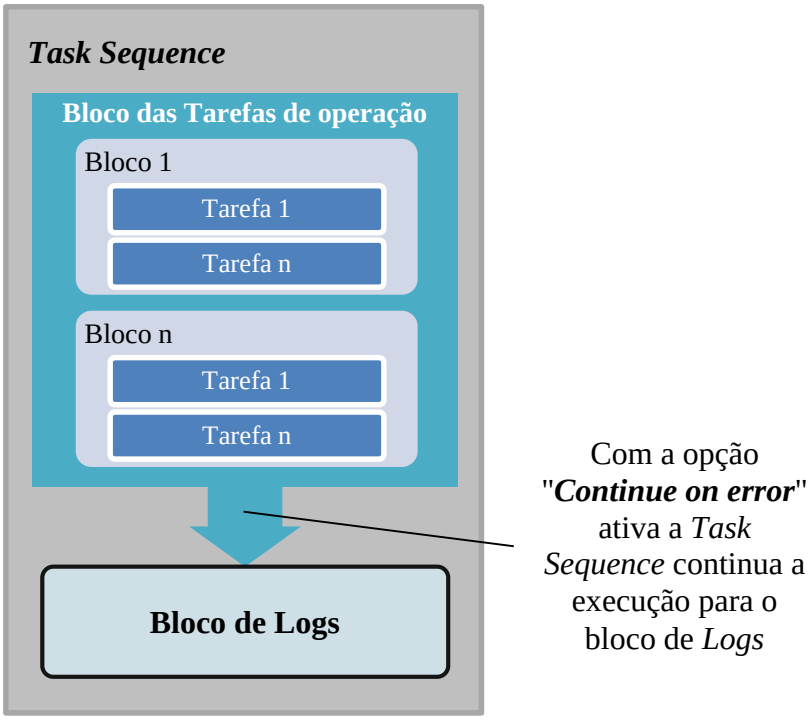


Figura 20 - Blocos principais da Task Sequence

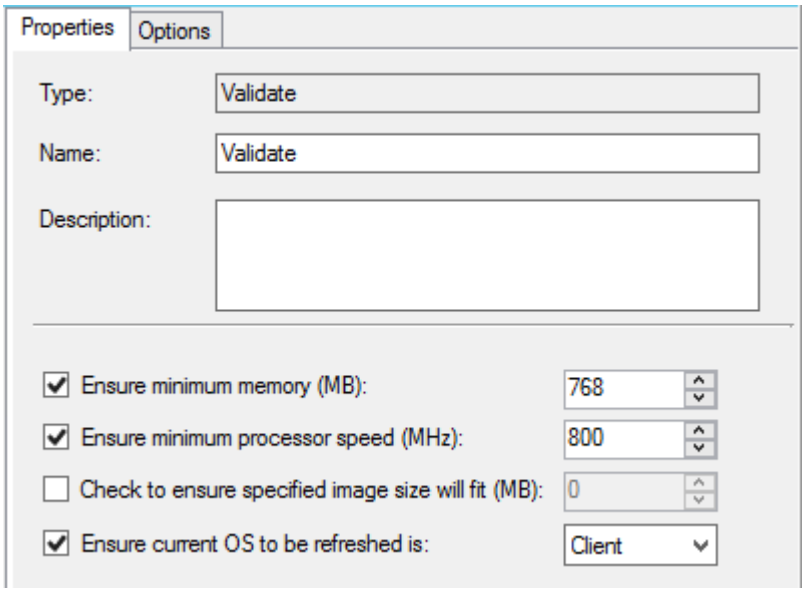
Esta informação poderá ser útil em casos em que o problema, ou problemas, ultrapassem o contexto dos *Drivers* e tenham a ver com outras situações, nomeadamente as relacionadas, por exemplo, com falhas de instalação do sistema ou do restante *Software* auxiliar. Devemos por esse motivo forçar que o Bloco de tarefas de operação continue em caso de erro, para garantir que temos esses *logs* disponíveis. Logo essa opção foi ativada para esse Bloco Principal no SCCM.

3.2.3.2 Tarefas desenvolvidas e implementadas

As tarefas desenvolvidas, implementadas e testadas que contribuíram para o funcionamento da solução foram:

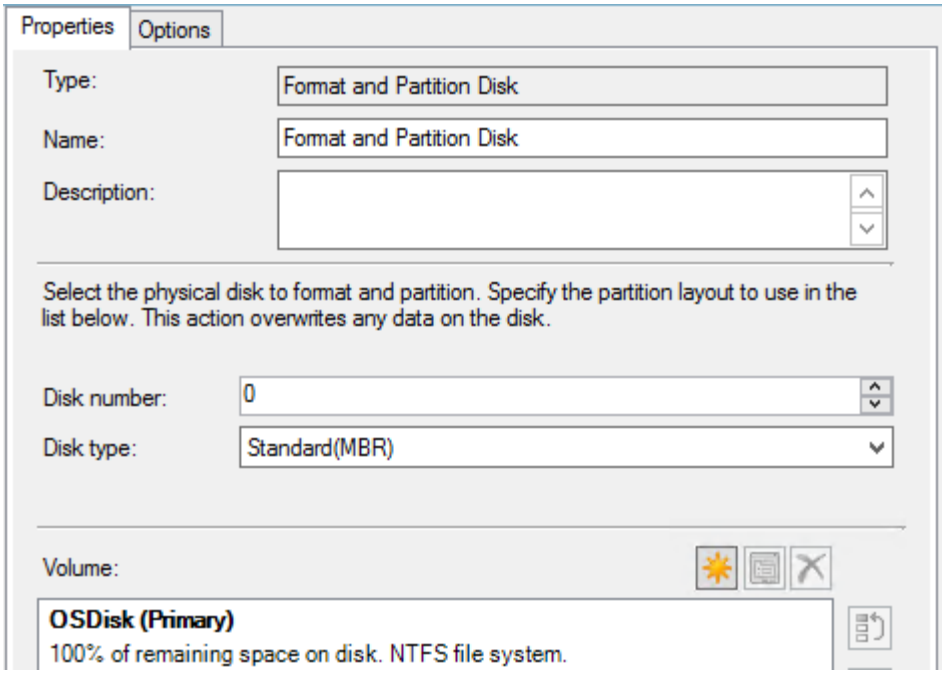
3.2.3.2.1 Validar condições de instalação da máquina cliente

Esta tarefa standard do SCCM permite verificar as condições de hardware base da máquina cliente, para validar a possibilidade de continuação da operação para as tarefas seguintes.



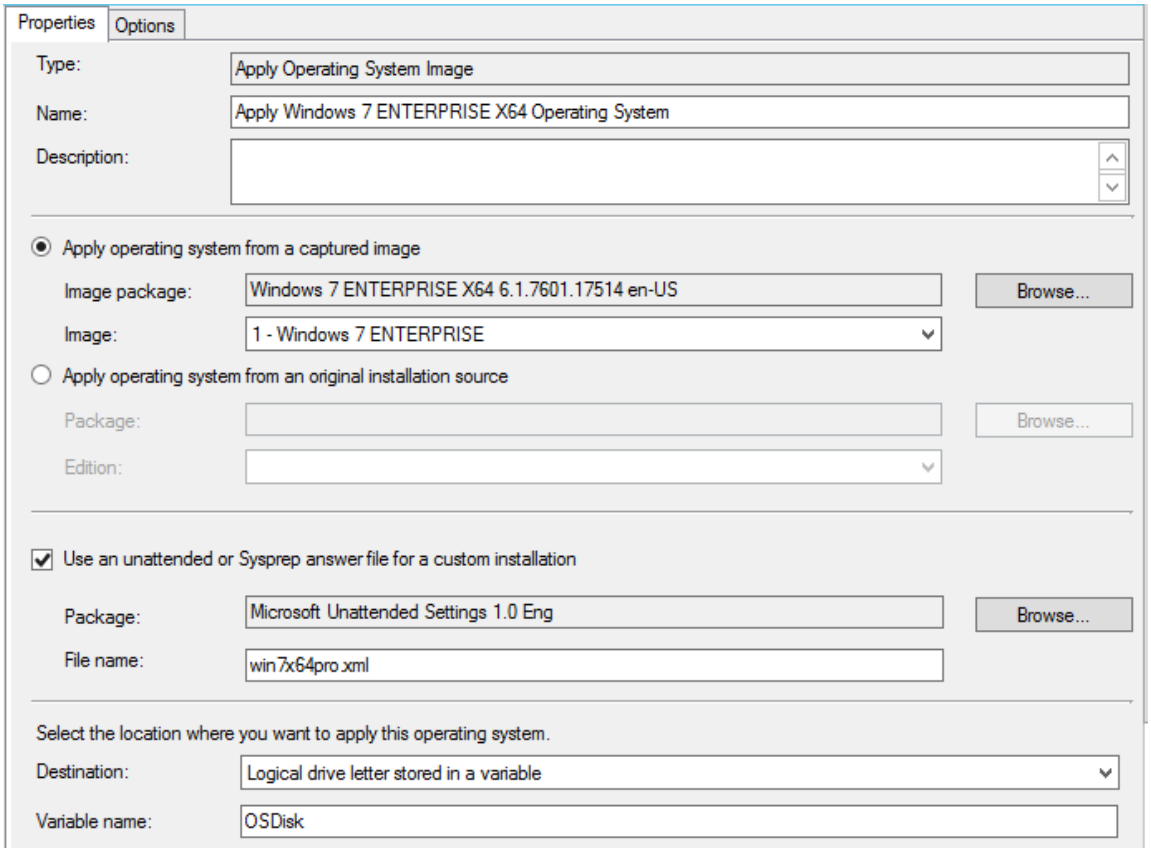
3.2.3.2.2 Formatar disco

O SCCM tem um tipo de Tarefa específica para esta função.



3.2.3.2.3 Instalar Sistema Operativo

Tal como para a formatação do disco, o SCCM tem um tipo de Tarefa específica para a instalação do Sistema Operativo, na qual temos de escolher as opções adequadas.



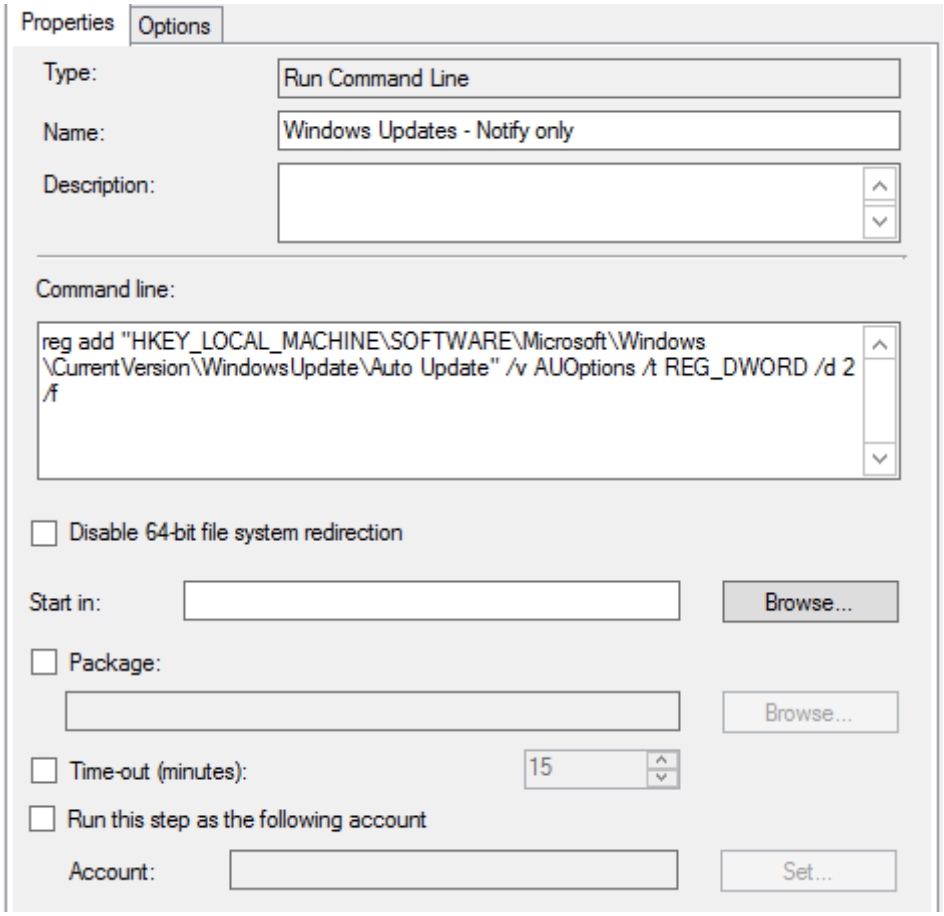
3.2.3.2.4 Desativar atualizações automáticas do Windows

É importante garantir que não são instaladas atualizações do *Windows* (que incluem *Drivers*) para nos assegurarmos de que vamos obter o melhor *Driver* (de preferência da página do fabricante), logo, a *Task* que se segue foi criada para este efeito, através da injeção do seguinte *Script* no campo "*Command Line*" de uma *Task* do tipo "*Run Command Line*":

```
Reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Windows Update\Auto Update" /v AUOptions /t REG_DWORD /d 2 /f
```

Este irá alterar o Registo do *Windows*, para impedir o mesmo de receber atualizações, ficando assim este serviço em modo "Apenas Notificação". Este modo é essencial para garantir que não vão ser instalados *Drivers* automaticamente enquanto são feitos os testes na máquina de referência, ao mesmo tempo que nos permite instalar os mesmos, se assim o desejarmos. Para esta ação ser efetiva, terá de ser realizada em conjunto com a equipa responsável pelas políticas, visto que estas se sobrepõem. Logo,

também terá de ser alterada a "Política de Atualizações do *Windows*", via Group Policy Object (GPO).

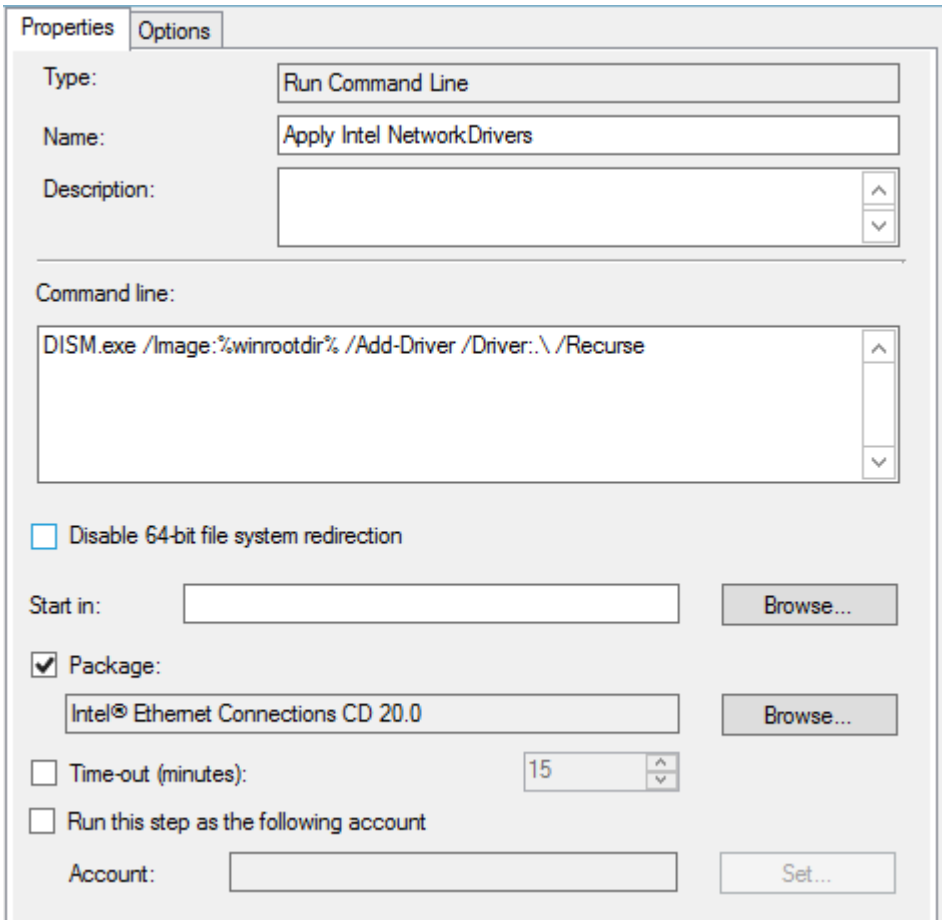


3.2.3.2.5 Instalar Drivers de rede

Os *Drivers* de rede e armazenamento têm de ser instalados logo no arranque, no *Windows PreInstallation Environment* (WinPE), sendo os restantes instalados depois, podendo ser instalados através do Agente do fabricante, numa fase posterior da instalação já com acesso à rede, mas com todo o processo sob controlo da *Task Sequence*.

A *Task* seguinte foi criada para instalar os *Drivers* das placas de rede da Intel (residentes na totalidade dos modelos do CLIENTE), para garantir o acesso à rede (de todos os modelos) e conseguir executar as tarefas seguintes por essa via, através do seguinte *Script*:

```
Dism.exe /Image:%winrootdir% /Add-Driver /Driver:.\/Recurse
```



3.2.3.2.6 Instalar a Framework .NET

Para o funcionamento do *Software* dos fabricantes das máquinas é necessário instalar a *Framework* .NET (é recomendada a 3.5 ou superior).

Ficou configurada a instalação silenciosa da versão 4.52, com geração de relatório ("*log*"), da seguinte forma [57]:

```
cmd /c NDP452-KB2901907-x86-x64-AllOS-ENU.exe /passive /norestart
```

Parâmetro	Descrição
/c	executa a operação especificada e pára.
/passive	Instalação Unattended, ou seja, sem interação, mas com apresentação de estado. Se houver necessidade de reiniciar no final da instalação, é apresentada uma caixa de diálogo com um aviso de reinício em 30 segundos.
/norestart	Não reinicia o computador quando a instalação terminar.

3.2.3.2.7 Instalar o Software do fabricante

No caso do *Software* para as máquinas cliente, não é necessário instalar. É sim efetuada a cópia do programa, via SCCM, para uma pasta na máquina de referência, injetando numa *Task* o seguinte comando:

```
cmd /c xcopy /E *.* "%SystemDrive%\ThinInstaller\"
```

Parâmetros usados	**Descrição**
/e	Copia todos os subdiretórios, mesmo que estejam vazios [1].

Para o *Software* da Lenovo ser instalado (ou copiado) apenas para as máquinas Lenovo, é necessário ainda introduzir a seguinte condição, através de uma *Query* WMI num statement if desta tarefa, na *Task Sequence*:

```
SELECT * FROM win32_ComputerSystem WHERE Manufacturer = “Lenovo”
```

3.2.3.2.8 Configurar o Software do fabricante

É necessário editar o ficheiro "*ThinInstaller*.exe.*Configuration*" na raiz desta pasta e indicar o *path* para o repositório (criado pelo *Update Retriever*). Este ficheiro é um XML e pode ser editado via *Powershell*.

O *Powershell* suporta totalmente a manipulação de ficheiros XML, cujo formato e opções a configurar são os seguintes:

```
<Configuration>
…
   <RepositoryPath>\\Driverserver\e\DriversFromManufacturers\' + $maker +
'\' + $Model</RepositoryPath>
   <LanguageOverride>EN</LanguageOverride>
   <ContentMode>Active</ContentMode>
…
</Configuration>
```

e desta forma podemos alterar todos os campos necessários para obter a configuração desejada, introduzindo numa *Task* específica para o efeito, o *Script*:

```
Powershell -ExecutionPolicy Bypass -Command "
$Maker=(get-content $env:SystemRoot\logs\ComputerManufacturer.log);
$Model=(get-content $env:SystemRoot\logs\ComputerModel.log);
[xml] $xml=get-content '%SYSTEMDRIVE%\ThinInstaller\ThinInstaller.exe.configu
ration';
```

```
 $xml.Configuration.repositorypath = '\\Driverserver\e\DriversFromManufacturers\' +  $maker + '\' + $Model;
$xml.Configuration.languageoverride = 'EN';
$xml.Configuration.contentmode = 'Test';
$xml.save('%SYSTEMDRIVE%\ThinInstal ler\ThinInstaller.exe.Configuration')"
```

Parâmetros	**Descrição**
-ExecutionPolicy	Define a *Execution Policy* para a sessão corrente e guarda-a na variável de ambiente $env:PS*ExecutionPolicyPreference*. Este parâmetro não altera a *Execution Policy* do *Powershell* que está definida no *registry*.
-Command	Executa os comandos especificados (e quaisquer parâmetros) como se fossem escritos na *command prompt* do *Powershell*, e depois sai (excepto se *NoExit* for especificado). O valor de *Command* pode ser "-", uma *String* ou um *Script block*. Se o valor de *Command* for "-", o texto do comando é lido do *standard input*. Os *Script* blocks têm de estar envoltos em chavetas ({}). Apenas podemos especificar um *Script block* ao correr o *Powershell*.exe. Os resultados do *Script* são returnados à *parent shell* como objetos XML deserializados. Se o valor de *Command* for uma *String*, *Command* tem se ser o último parâmetro, porque quaisquer caracteres após *Command* são interpretados como argumentos de *command*. Para escrever uma *String* que corre um *Powershell command*, usa-se o formato: "& {<*command*>}" onde as aspas indicam uma *String* e o *invoke operator* (&) faz com que o comando seja executado.

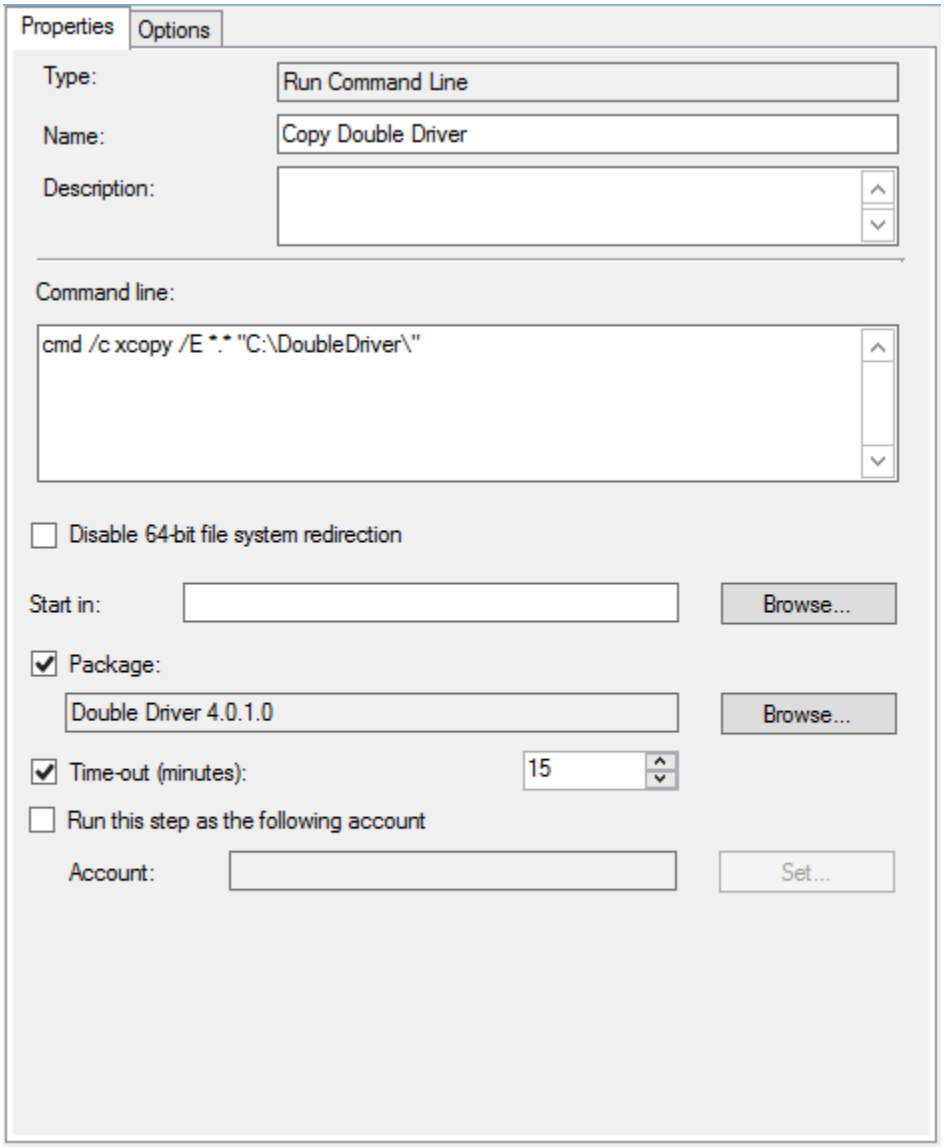
Este *Script* foi desenvolvido (neste exemplo, para o caso da Lenovo), para alterar os campos necessários do XML que contém as configurações, para as opções que pretendermos e guardar o mesmo no final.

*To*dos os procedimentos mencionados têm o seu equivalente para o caso dos modelos HP.

3.2.3.2.9 Instalar o Double Driver

No caso do *Software Double Driver*, tal como o software do fabricante não é necessário instalar. Também é efetuada a cópia do programa, via SCCM, para uma pasta na máquina de referência, injetando numa *Task* o seguinte comando:

```
cmd /c xcopy /E *.* "%SystemDrive%\DoubleDriver\"
```



3.2.3.2.10 Correr Software do fabricante

Para este efeito é necessário definir uma outra *Task*, para executar o *Software*, que conterá o seguinte *Script* (exemplo para o caso da Lenovo):

```
cmd /c %SYSTEMDRIVE%\ThinInstaller\ThinInstaller.exe /CM -search A -action
INSTALL -repository '\\Driverserver\e\DriversFromManufacturers\' + (type
%windir%\logs\ComputerManufacturer.log) -noicon -includerebootpackages 1,3,4
```

Este instala todos os pacotes críticos, recomendados e opcionais sem apresentar notificação. Os pacotes que impliquem reiniciar o sistema, vão forçar o mesmo a reiniciar.

Descrição das opções:

-search

- C - Atualizações Criticas
- R - Atualizações Criticas e Recomendadas
- A - Atualizações Criticas, Recomendadas e Opcionais (todas)

-action

- INSTALL - Instala as atualizações
- LIST - Notifica sobre disponibilidade de atualizações

-noicon (permite desligar o tooltip que aparece quando são encontrados ou é feito o *Download* ou instalação de *Packages*)

-includerebootpackages 1,3,4

1 - *Reboot* forçado pelo *package*;

3 - *Reboot* necessário para ter efeito; O *Software* vai forçar o *reboot* após a instalação de todos os *Packages*;

4 - O *package* força o shutdown.

-*NoReboot* (Previne o *reboot* após a instalação de um *package* do tipo 3, não impedindo para um *package* de tipo 1 ou 4)

O *Thin Installer* procura sempre pelo ficheiro "*database*.xml" na pasta definida como Repositório. Este ficheiro associa cada tipo de máquina, sistema operativo e língua com os *Packages* de atualização. Na falta deste ficheiro, o *Thin Installer* assume que todos os *Packages* no Repositório são candidatos (são aplicáveis à máquina cliente).

Se o repositório for criado com o *Update Retriever*, este ficheiro é criado na pasta lá definida.

3.2.3.2.11 Obter informação do Hardware

Para a automatização das operações, são necessárias várias informações sobre os vários modelos de *Hardware* e Sistemas Operativos usados. O *Powershell* em combinação com o WMI, vão permitir obter essas informações (que irão ajudar nas decisões da automação).

Para obter a informação necessária sobre cada modelo, foi desenvolvido o seguinte *Script*, ao qual foi atribuído o nome *IdentifyFaultyDrivers*.ps1:

```
Powershell -ExecutionPolicy Bypass -Command "& {

Set-Location $Env:SystemRoot\logs;
```

```
(Get-WmiObject Win32_ComputerSystem).Model -replace '\/', '-' | Out-File
ComputerModel.log;

(Get-WmiObject Win32_ComputerSystem).name  | Out-File
$Env:SystemRoot\logs\ComputerName.log;

(Get-WmiObject Win32_ComputerSystem).manufacturer.Split(' ')[0] | Out-File
ComputerManufacturer.log;

-join(Get-WmiObject Win32_OperatingSystem).name.Split('|')[0].Split('
')[1..2] | Out-File OSName.log;

if ((Get-WmiObject Win32_OperatingSystem).OSArchitecture -eq '64-bit'){'x64'
| Out-File OSArchitecture.log}else{'x86' | Out-File OSArchitecture.log};

(Get-WmiObject Win32_OperatingSystem).version | Out-File OSVersion.log;

}"
```

Este *Script* foi introduzido na *Task Sequence* de captura de *Drivers* no SCCM, numa *Task* que foi denominada por "*Store machine info.cmd*"

No SCCM o comando é lançado via Linha de comando, logo, temos de executar o *Powershell* a partir desta, usando o comando que executa o mesmo, juntamente com a definição da política de execução necessária para correr os comandos internos (*cmdlets*) suficientes para obter todas as informações que precisamos:

```
Powershell -ExecutionPolicy Bypass -Command "& {Cmdlets do Powershell}”
```

Em termos de políticas de execução, podemos usar a *-ExecutionPolicy* com o parâmetro *Bypass*. Desta forma, o cliente do *Configuration Manager* contorna a configuração do *Powershell* no computador cliente e os *Scripts* não assinados podem correr.

3.2.3.2.12 Verificar informação do estado dos dispositivos de Hardware

Uma das funcionalidades chave que este processo implica é uma forma de testar o funcionamento dos dispositivos. Para detetar erros relativos a *Drivers* (instalação e funcionamento) recorreu-se a uma verificação automática da mesma informação disponibilizada no Gestor de dispositivos do *Windows* (“*Device Manager”*) recorrendo a *Powershell* e WMI.

O *Device Manager* é uma ferramenta de troubleshooting importante que fornece uma vista centralizada e organizada de todos os dispositivos de *Hardware* instalados e permite-nos identificar conflitos entre eles, gerir os seus *Drivers*, alterar as opções de configuração de *Hardware* e até ligar ou desligar componentes específicos de *Hardware*.

Os dispositivos desconhecidos aparecem no *Device Manager* quando o *Windows* não consegue identificar um componente e fornecer-lhe um *Driver*. Um dispositivo desconhecido não é apenas desconhecido; não funciona até instalarmos um *Driver* correto.

O *Windows* consegue identificar a maioria dos dispositivos e obter os respetivos *Drivers* automaticamente. Quando este processo falha, ou desligamos a obtenção automática dos *Drivers*, teremos de identificar o dispositivo e obter o *Driver* por nossa conta.

A listagem de cada dispositivo no *Device Manager*, contém informação detalhada do *Driver*, recursos do sistema e outras configurações. Quando alteramos uma configuração de um componente de *Hardware*, é alterada a forma como o *Windows* trabalha com esse *Hardware*.

Podemos consultar as seguintes informações dos dispositivos [58]:

- ID de instância do dispositivo;
- ID de *Hardware*;
- IDs compatíveis;
- ID de dispositivo correspondente;
- Serviço;
- Enumerador;
- Capacidades;
- Sinalizadores Devnode;
- ConfigFlags;
- CSConfigFlags;
- Relações de ejeção;
- Relações de remoção;
- Relações de barramento;
- Filtros superiores do dispositivo;
- Filtros inferiores do dispositivo;
- Filtros superiores da classe;
- Filtros inferiores da classe;
- Instalador para classes;
- Coinstaladores para classes;
- Coinstaladores para dispositivos;
- Revisão *firmware*;
- Estado atual de energia;

- Funções de energia;
- Mapeamentos de estado de energia.

Nota: Nem todas as propriedades descritas anteriormente aparecem associadas a cada dispositivo.

Embora rico em funcionalidades, o *Device Manager* está apenas disponível numa versão com Interface gráfica, não correndo em modo consola e, portanto, não permitindo construir operações automatizadas que dependam deste.

Para esse efeito foi desenvolvido um *Script* em *Powershell*, que recorre ao WMI, analisando a mesma informação disponibilizada pelo *Device Manager*, mas de forma automatizada.

O WMI tem uma classe - Win32_PNPEntity - que retorna informação acerca dos dispositivos encontrados no Gestor de Dispositivos ("*Device Manager*"). Existe uma propriedade nesta classe - Config*Manager*ErrorCode - que diz se o dispositivo está a funcionar corretamente ou não. Se o valor de Config*Manager*ErrorCode é 0, então o dispositivo está a funcionar. Se o Config*Manager*ErrorCode for algum valor que não *Zero*, então algo estará errado.

Portanto, para escrever um *Script* que retorne uma lista de dispositivos que não estão a funcionar corretamente, é apenas necessário ligar à classe Win32_PNPEntity e fazer uma *Query* por todos os dispositivos com um Config*Manager*ErrorCode diferente de 0.

O *Powershell* tem comandos que fazem a interação com o WMI, logo se quisermos "perguntar" à classe Win32_PNPEntity por dispositivos que não estão configurados, basta escrever:

```
Get-Wmi<i>Object</i> Win32_PnpEntity | Where-<i>Object</i>{$_.Config<i>Manager</i>ErrorCode -gt 0}
```

Quanto aos erros que poderiam surgir, foi necessário fazer um levantamento dos possíveis erros relativos aos dispositivos [59][60][61], tendo resultado o *Script* que se apresenta no Anexo 1.

Exemplo da informação obtida com este *Script*, para o caso de um dispositivo desativado:

Device Name	HardwareID	Error Code	ErrorDescription
Intel(R) Centrino (R) Wireless Bluetooth(R) 3.0 + High Speed Adapter	USB\VID_8086&PID_0189&REV_6919 USB\VID_8086&PID_0189	22	Este dispositivo está desativado. Ativar o dispositivo.

Poderão ainda existir (com grande probabilidade), *Drivers* genéricos, instalados. Para os identificar, basta procurar pelos nomes que incluem "*Standard*", "*Generic*" ou mesmo "*Microsoft*". É conveniente verificar se existem *Drivers* do fabricante disponíveis, mas se não existirem, estes poderão ser considerados, como solução de recurso.

Para saber os *Drivers* "Genéricos", podemos recorrer ao *Powershell* e WMI, da seguinte forma:

```
Get-WmiObject Win32_PNPEntity | Where-Object{$_.Name –Match "Generic"} |
Select Name, HardwareID
```

```
Name               HardwareID
----               ----------
Generic USB Hub    {USB\VID_8087&PID_0024&REV_0000, USB\VID_8087&PID_0024}
```

Para o caso dos *Drivers* "*Standard*" (neste exemplo "Padrão"):

```
Get-WmiObject Win32_PNPEntity | Where-Object{$_.Name –Match "Padrão"} |
Select Name, HardwareID
```

```
Name                HardwareID
----                ----------
Teclado Padrão      {ACPI\PNP0303, *PNP0303}
PS/2
```

E para os "*Microsoft*":

```
Get-WmiObject Win32_PNPEntity | Where-Object{$_.Name -Match "Microsoft"} |
Select Name, HardwareID
```

```
Name                         HardwareID
----                         ----------
Controlador BIOS Microsoft   {ROOT\mssmbios}
System Management
Transformador Microsoft      {ACPI\ACPI0003, *ACPI0003}
Interface de gestão          {ACPI\PNP0C14, *PNP0C14}
Microsoft Windows para
ACPI
Placa Microsoft ISATAP       {*ISATAP}
Sistema compatível com       {ACPI_HAL\PNP0C08, *PNP0C08}
Microsoft ACPI
Placa Miniport WiFi          {{5d624f94-8850-40c3-a3fa-
Virtual da Microsoft         a4fd2080baf3}\vwifimp}
Bateria Composta da          {COMPOSITE_BATTERY}
Microsoft
```

Juntando tudo num mesmo *Script*, obtemos o código do Anexo 2.

3.2.3.2.13 Verificar assinatura digital dos Drivers

Em caso de *Drivers* em falta ou que originaram algum erro e, consequentemente, terão se ser reinstalados, na execução do processo, onde procurar *Drivers* alternativos? Apenas no fabricante e *Microsoft*? Outras fontes?

Neste caso, e se for mesmo necessário recorrer a outras fontes, temos de garantir que os *Drivers* estão assinados, porque o CLIENTE só aceita *Drivers* assinados, logo é usada a ferramenta *Microsoft Signtool*, para efetuar essa verificação (Anexo 3):

3.2.3.2.14 Testar Drivers (testes de stress com o Microsoft Verifier)

Para complementar, optou-se por verificar ainda o seu comportamento, através de testes de *stress* adicionais, recorrendo para esse efeito à ferramenta *Microsoft Verifier*, da seguinte forma:

```
Cmd /c %WINDIR%\System32\Verifier /standard /faults /all
```

Desta forma são configuradas as definições de teste pretendidas e será necessário forçar um *Restart* na *Task Sequence*, para o programa correr, quando a máquina reiniciar.

Se quisermos verificar apenas alguns *Drivers* são necessários os nomes completos dos ficheiros:

```
/Driver DriverList
```

Sendo *Driver*List a lista de *Drivers* em nome binário, como *Driver*.sys, separados por espaço, ficando o comando com o seguinte aspecto:

```
Cmd /c %WINDIR%\System32\Verifier /standard /faults /Driver1.sys Driver2.sys
```

Para excluir alguns da análise; caso estejam todos selecionados com /*all* é apenas necessário acrescentar:

```
/Driver.exclude DriverList
```

O ficheiro *DUMP* gerado pelo *Driver Verifier* (em caso de problemas) ficará armazenado em: %*SystemRoot*%\MEMORY.DMP

Após o programa correr (foi analisado o tempo necessário, para introduzir uma pausa na *Task Sequence*, por forma a garantir que há tempo suficiente para o programa correr antes de passar à próxima tarefa), é necessário fazer um Reset às configurações, por forma a evitar que o mesmo continue a correr a partir do próximo reboot, logo são adicionadas mais duas tarefas:

Repor configurações iniciais (após reiniciar, já não irá testar os *Drivers*):

```
Cmd /c %WINDIR%\System32\Verifier /reset
```

E *Restart*, para garantir que a partir deste momento é possível correr a ferramenta *Double Driver*, para fazer o backup dos *Drivers* instalados na máquina referência (em caso de não existirem problemas).

3.2.3.2.15 Correr Double Driver para backup dos Drivers para o repositório

Condição de execução:

```
WMI Query SELECT * FROM Win32_PNPEntity WHERE ConfigManagerErrorCode != 0
```

A condição definida faz com que o *Software* só faça o *backup* dos *Drivers* para o Repositório do SCCM (para distribuição), no caso de não haver problemas identificados nos dispositivos e, nesse caso, executará então o *Script*:

```
cmd /c %SystemDrive%\DoubleDriver\ddc b /target:\\DriversRepository\...
```

3.2.3.2.16 Analisar DUMP (Microsoft Windows Debugger)

Em caso de erros, foi gerado o ficheiro *DUMP* respetivo no passo anterior, que poderá ser analisado com a ajuda desta ferramenta.

Na Command line, para abrir um ficheiro *DUMP*, quando lançamos o *Windbg*, basta fazer [62]:

```
 Windbg -z DumpFileName
```

Ou seja:

```
Cmd /c %ProgramFiles%\Debugging Tools for Windows (x86)\Windbg.exe" -z
%SystemRoot%\MEMORY.DMP
```

A extensão **!analyze** apresenta informação acerca da "*current exception*" ou "*bug check*" [63]:

```
!analyze –v
```

A opção -v (verbose *mode*) também é útil, para obtermos informação detalhada.

No entanto, devido à significativa complexidade deste tema, esta análise ficou fora do âmbito deste trabalho, mas fica a ideia, que é válida e poderá ser trabalhada para o efeito no futuro.

3.2.3.2.17 Importar Drivers para a Task Sequence

Para adicionar cada modelo à *Task Sequence* (a um bloco de tarefas que permita a cópia, em primeiro lugar para uma *Task Sequence* de Testes de Qualidade/*Change Management* e depois, após aprovação, para a *Task Sequence* da *Master*, onde serão então distribuídos para as máquinas cliente) foi necessário desenvolver um *Script*, em,

que usa os *cmdlets Powershell* específicos para o SCCM, já apresentados na descrição do *Powershell*, criando automaticamente a informação necessária para posterior *Deployment* dos *Drivers*, para cada modelo, sem necessidade de executar manualmente todas as tarefas necessárias para o fazer, algo morosas e repetitivas.

Antes de mais é necessário obter a informação para as variáveis que serão usadas:

```
$CMSiteCode = $([WmiClass]"\\ConfigPR01\Root\ccm: SMS_client").getassigned-
site() | Select sSiteCode

$CMSiteServer = 'server.domain.local'

$CMNameSpace = 'root\SMS\site_$CMSiteCode'

$DriverPackageName = $manufacturer + ' ' + $model

$DriverSource =
'"\\DriverServer\e\DriversRepository\$OSName\$OSArchitecture\$manufacturer
$model"

$DriverPackageSource = '\\DriverServer\e\Sources\DriverPackages\' +
$manufacturer + ' ' + $model + ' - ' + $OSName + ' ' + $OSArchitecture

$DriverCategoryName01 = $OSName + ' ' + $OSArchitecture

$DriverCategoryName02 = $manufacturer + ' ' + $model
```

Para podermos usar os *cmdlets* é necessário importar o módulo do *Configuration Manager*, o que pode ser feito da seguinte forma:

```
Import-Module (Join-Path $(Split-Path $env:SMS_ADMIN_UI_PATH)
ConfigurationManager.psd1)
```

De seguida podemos criar o *Driver package*:

```
New-CMDriverPackage -Name $DriverPackageName -Path $DriverPackageSource -
PackageSourceType storageDirect -Verbose
$DriverPackage = Get-CMDriverPackage -Name $DriverPackageName
```

E distribuir o *Driver Package* criado para o(s) *Distribution Point(s):*

```
Start-CMContentDistribution -DriverPackageName $DriverPackage.Name -
DistributionPointName $CMSiteServer -Verbose
```

É ainda necessário criar Categorias Administrativas, para organizar os *Drivers*:

```
If ((Get-CMCategory -Name $OS) -eq $null)
{
    New-CMCategory -CategoryType DriverCategories -Name $OS
}

$DriverCategory1 = Get-CMCategory -Name $OS

If ((Get-CMCategory -Name $ModelName) -eq $null)
{
```

```
    New-CMCategory -CategoryType DriverCategories -Name $ModelName
}

$DriverCategory2 = Get-CMCategory -Name $ModelName

$DriverCategories = @()
$DriverCategories += $DriverCategory1
$DriverCategories += $DriverCategory2
```

Agora é necessário obter os *Drivers* da fonte:

```
$Drivers = Get-ChildItem -Path $DriverSource -Include *.inf –Recurse
```

E, finalmente, importar os *Drivers* para o SCCM:

```
foreach ($Driver in $Drivers)
{
    Import-CMDriver -UncFileLocation $Driver.FullName `
    -DriverPackage $DriverPackage `
    -EnableAndAllowInstall $true `
    -AdministrativeCategory $DriverCategories `
    -ImportDuplicateDriverOption AppendCategory `
    -ErrorAction SilentlyContinue `
    -Verbose
}
```

3.2.4 Testes, resultados e avaliação da solução

3.2.4.1 *Testes efetuados*

Foram efetuados vários testes, com os três modelos disponíveis em Portugal, que consistiram no seguinte:

- Foi efetuada a instalação pelo método anteriormente em uso (manual), em que após o provisionamento do sistema operativo e das aplicações é necessário correr e configurar as mesmas manualmente, não só no Servidor (*Update Retriever*), mas também nas Máquinas de Referência (*Thin installer* e *Double Driver*);
- Foi efetuada a instalação com a solução desenvolvida, em que após o provisionamento do sistema operativo e das aplicações não é necessário correr e configurar as mesmas manualmente, visto que este processo foi programado e a execução do lado do cliente está totalmente automatizada.

Para os testes realizados, foi medido o tempo médio necessário para concluir a operação, recorrendo a vários elementos da equipa, cada um com um grau de experiência diferente nesta matéria.

3.2.4.2 *Testes não efetuados*

Não foram efetuados testes com os modelos HP, pois são *desktops* e não estão disponíveis em Portugal, visto que o CLIENTE apenas disponibilizou alguns portáteis, e estes são todos da Lenovo. Por esse motivo, e apesar de terem sido analisadas as ferramentas deste fabricante e confirmada a possibilidade de as usar, no âmbito da solução desenvolvida, as mesmas não são apresentadas em qualquer exemplo, por não ter sido possível testar o seu funcionamento.

Não foi também analisado o comportamento da solução com *Drivers* provenientes de fontes mais duvidosas, pois, nos casos em que o primeiro ciclo não resultou na instalação de todos os *Drivers*, os *Drivers* identificados como estando em falta, apesar de não serem instalados pelas aplicações do fabricante, encontravam-se disponíveis no Website do mesmo, pelo que passaram na verificação da assinatura e também não apresentaram problemas nos testes de *stress* pelo *Microsoft Verifier*. Contudo, bastava, para esse efeito, ser introduzido um *Driver* proveniente de outras fontes.

3.2.4.3 *Resultados obtidos*

Conclui-se que a solução reduz em pelo menos 50% o tempo necessário para obter os *Drivers* a depositar no repositório, para posterior distribuição.

Se pelo método manual podemos demorar mais de duas horas para finalizar o processo, no método otimizado conseguimos concluir o mesmo em cerca de uma hora (menos, no caso do modelo equipado com um SSD), no caso de não haver faltas ou erros em *Drivers*, pois caso tal aconteça, há que somar o tempo para os encontrar e adicionar.

A maior vantagem desta solução é evitar termos de identificar manualmente todos os dispositivos e procurar e instalar manualmente todos os *Drivers* para cada modelo (o que é bastante moroso). Temos então um primeiro ciclo, que pode permitir instalar todos os *Drivers*, sem qualquer trabalho adicional (como aconteceu com um dos modelos) e, além disso, receber a informação de faltas ou erros (nomeadamente a descrição do problema e o *Hardware* ID), devidamente estruturada, para facilitar um eventual ciclo posterior, de pesquisa de *Drivers* em falta e/ou resolução de possíveis problemas com *Drivers* já instalados no primeiro ciclo.

Após esse primeiro ciclo ser executado, conseguimos então ter os *Drivers* instalados e a informação dos que não existem ou têm problemas, podendo agir em conformidade. Esta acção, a ser necessária, também é mais célere, pois rapidamente conseguimos encontrar um *Driver* substituto (se este existir) e colocá-lo no share para ser instalado.

De notar que após encontrarmos o(s) *Driver*(s) que estavam em falta, temos de executar um *script* (constante n Anexo 4) que instala o(s) *Driver*(s) que tivermos colocado numa pasta específica, que podemos criar dentro da pasta do modelo onde podemos colocar os *Drivers* que tivemos de procurar manualmente.

Isto pode ser "enviado" ao agente do SCCM instalado na máquina de referência, para ser executado nessa máquina, pelo que de seguida podemos ainda definir uma nova execução da verificação (o *script* desenvolvido para obter informação do *Hardware*), pela mesma via, para garantir que não são reportados mais problemas e que temos de facto instalados os *Drivers* para todos os dispositivos. Depois devemos ainda executar novamente o *script* para criar o *package* e importá-lo para a *Task Sequence*, já com o(s) novo(s) *Driver*(s) incluído(s).

3.2.4.4 *Avaliação*

Analisando os resultados obtidos, podemos concluir que além da poupança em termos de tempo ser só por si considerável e justificar o tempo investido na construção da solução, este processo também evita a deslocação obrigatória de um dos elementos da equipa, que se dedicava maioritariamente a este procedimento, permitindo uma redução de custos significativa, por ambos os motivos.

Justificar-se-á por isso a continuidade do desenvolvimento da mesma, por forma a permitir a sua utilização em qualquer projeto do mesmo tipo, visto que o retorno a longo prazo, compensará o investimento efetuado e aumentará certamente a satisfação dos clientes, pois possibilitará, não só um melhor serviço, como, eventualmente, uma redução no preço final ao cliente.

Capítulo 4

Conclusão

Neste capítulo são apresentados: um sumário do trabalho realizado, as dificuldades encontradas, as conclusões e possibilidades de trabalho futuro.

4.1 Trabalho desenvolvido

Este projeto teve como finalidade a otimização de parte do processo de aprovisionamento de sistemas operativos, mais concretamente do processo de captura dos *Drivers*, que são posteriormente distribuídos para todas as máquinas cliente físicas, utilizadas pelos colaboradores da Organização.

Na prática o Sistema Operativo é instalado na máquina de referência, as aplicações necessárias são instaladas e executadas de forma automática e não manual, os scripts são executados e a informação obtida é passada para o servidor, toda de uma vez e sem ser necessário estar fisicamente presente, nem procurar manualmente e caso a caso, no Gestor de Dispositivos, pela identificação dos dispositivos problemáticos, o que é bastante moroso.

Esta informação pode depois ser facilmente analisada no servidor, ao qual a equipa tem acesso remoto, e fornecidos ou substituídos os *Drivers* em falta ou com problemas respetivamente, que foram automaticamente identificados pelo processo descrito.

Se não houver qualquer problema identificável, o processo captura com sucesso todos os *Drivers* necessários, coloca-os no repositório e cria os campos necessários na *Task Sequence* para os instalar posteriormente.

4.2 Dificuldades encontradas

Uma das principais dificuldades foi o facto de todo o universo das tecnologias e ferramentas ser uma absoluta novidade, desde o *Windows Server* 2012 até ao *Powershell*, passando pela maioria dos conceitos envolvidos.

A dimensão do projeto e de tudo o que envolve foi um desafio igualmente significativo, pois foi bastante complicado e moroso de absorver. A documentação é extensa e não muito fácil de entender, principalmente para alguém sem qualquer experiência na área, nem em projetos de menor dimensão, nem sequer académica.

O facto de estar dependente de terceiros e de sistemas complexos, começando pelo LAB usado para testes, também dificultou o processo. Além disso este servidor onde o LAB estava implementado manifestou desde sempre alguns problemas de *Hardware*, nomeadamente de discos, agravado pelo facto de não ter uma UPS e por vezes faltar a energia, o que causou alguns problemas no funcionamento e consequentemente atrasos adicionais que também contribuíram para retardar o processo, principalmente por não conseguir realizar todos os testes quando necessário e com o *timing* ideal e também por ter perdido por vezes algum do trabalho já realizado, que ainda não se encontrava nos backups.

4.3 Conclusões

A solução acaba por ser bastante simples, ou aparentemente simples, mas resolve o desafio colocado de uma forma expedita e eficaz.

O facto de a instalação e execução das ferramentas ser praticamente toda automatizada do lado do cliente, não só reduz a carga de mão-de-obra necessária, como diminui a necessidade de deslocações tão frequentes ao CLIENTE (pelo menos por este motivo), permitindo uma poupança considerável em termos de custos do projeto.

Pelos testes que foi possível efetuar, este processo reduzirá o tempo necessário para instalar cada máquina de referência em mais de 50% (no entanto seria necessário fazer mais testes, e com mais modelos, para obter valores mais apurados).

4.4 Trabalho Futuro

O funcionamento do *Verifier* necessitaria de mais testes, pois em caso de não originar um *Blue screen* teremos de contabilizar quanto tempo teríamos de dar na *Task Sequence* até passar à tarefa seguinte, ou seja, teríamos de saber o valor máximo que a *Task Sequence* devia esperar até forçar um *restart* e voltar a alterar as configurações do *Verifier* por forma a sair do modo de testes; na prática, desligar o *Verifier* para finalmente copiar os *Drivers* para o Repositório.

No caso de haver um *Blue screen*, é gerado um ficheiro com localização conhecida: %*SystemRoot*%\MEMORY.DMP, pelo que conseguimos perceber que o *Verifier* já terminou a sua execução, pela existência desse ficheiro e, neste caso, saber que será necessária uma intervenção manual, para fazer a análise do mesmo e tentar obter pistas

sobre qual o dispositivo a causar problemas e tentar obter um *Driver* melhor. De notar que este procedimento é suplementar e só acontece quando não foram detetados mais problemas na consulta à informação dos dispositivos via *Powershell*/WMI.

A análise do ficheiro *DUMP*, constatou-se ser de alguma complexidade, pelo que ficou excluído do âmbito deste trabalho, podendo, no entanto, justificar-se um aprofundamento do tema para aplicação futura, ficando por esse motivo referenciada.

Para generalizar a aplicação da solução, seria útil desenvolver mais um *Script Powershell* que substitua as ferramentas dos fabricantes e que seja genérico, ou seja, que funcione para qualquer fabricante, baseando a solução nos ficheiros .inf e não em aplicações disponibilizadas por cada um deles.

Neste projeto as ferramentas dos fabricantes existem para as marcas em questão e funcionam bem integradas na solução, sendo executadas por linha de comando com os parâmetros próprios, que permitem a automatização necessária para o que se pretende atingir, logo não era essencial, mas pode haver fabricantes que não disponibilizem estas ferramentas. Basicamente será um *Script* que faça a leitura do repositório e verifique os *Drivers* lá colocados, para cada modelo, e os instale, recorrendo aos ficheiros ".inf".

Seria ainda útil desenvolver uma interface gráfica, tipo formulário, para introdução dos campos que podem variar de projeto para projeto.

Anexo 1

Script para verificar informação do estado dos dispositivos de _Hardware_/_Drivers_:

```
#Obter informação sobre o modelo e SO

$manufacturer=(Get-WmiObject Win32_ComputerSystem).manufacturer.Split(' ')[0]

$Model = (Get-WmiObject Win32_ComputerSystem).Model -replace '\/', '-';

$OSArchitecture = if ((Get-WmiObject Win32_OperatingSystem).OSArchitecture -
eq '64-bit'){'x64'} else {'x86'};

$OSVersion = (Get-WmiObject Win32_OperatingSystem).version;

$OSName = -join(Get-WmiObject Win32_OperatingSystem).name.Split('|')[0].Split
(' ')[1..2];

$ComputerName = (Get-WmiObject Win32_ComputerSystem).name;


#Criar pasta para o modelo, apenas se ainda não existir (esta pasta servirá
tanto para os Drivers, como para os logs a analisar de cada modelo)

#Definir a localização corrente para esta pasta

Set-Location (Mkdir -p "\\Driverserver\e\DriversRepository\$OSName\$OSArchite
cture\$manufacturer $Model" -Force)


#Procurar dispositivos cujo valor do ConfigManagerErrorCode seja maior que 0
(ou seja, que tem algum tipo de problema).

$FaultyDevices = Get-WmiObject Win32_PnpEntity | Where-Object{$_.ConfigManage
rErrorCode -gt 0}

ForEach($FaultyDevice in $FaultyDevices)

{

   $ErrorDesc = Switch ($FaultyDevice.ConfigManagerErrorCode)

    {

      1 {“Este dispositivo não está configurado corretamente. Atualize o
      Driver.”}

      2 {“O Windows não consegue carregar o Driver para este dispositivo.
      Contacte o fabricante para obter uma atualização da BIOS ou atualize o
      Driver.”}

      3 {“O Driver para este dispositivo pode estar corrompido ou o sistema
      pode estar com pouca memória ou outros recursos. Feche alguns
      aplicativos abertos. Desinstale e reinstale o Driver. Instale RAM
      adicional.”}

      4 {“O Driver para este dispositivo ou o registo do Windows podem estar
      corrompidos. Atualize o Driver.”}

      5 {“O Driver para o dispositivo requer um recurso que o Windows não
      pode gerir. Atualize o Driver.”}
```

6 {“Outro dispositivo está a utilizar os recursos que este dispositivo necessita. Desligue o computador e mude os recursos para este dispositivo.”}

7 {“Os *Drivers* para este dispositivo precisam ser reinstalados.”}

8 {“O carregador de dispositivos (DevLoader) não foi encontrado. Por exemplo, o arquivo .inf pode referir-se a um arquivo ausente ou inválido. Atualize ou Reinstale o *Driver*.”}

9 {“Este dispositivo não está a funcionar corretamente porque a BIOS reporta os recursos para o dispositivo de forma incorreta. Contacte o fabricante para obter uma atualização da BIOS ou atualize o *Driver*.”}

10 {“Este dispositivo não pode iniciar. O dispositivo não está presente, não funciona corretamente ou não possui todos os *Drivers* instalados. Atualize o *Driver*.”}

11 {“O *Windows* parou de responder ao tentar iniciar este dispositivo, portanto não tentará iniciá-lo novamente. Atualize o *Driver*.”}

12 {“Este dispositivo não encontrou recursos livres suficientes para usar. Desative o dispositivo conflituante.”}

13 {“O *Windows* não consegue verificar os recursos do dispositivo. O dispositivo não está presente, não está a funcionar corretamente ou não possui todos os *Drivers* instalados. Use 'Detetar *Hardware*'.”}

14 {“Este dispositivo não pode funcionar corretamente até que se reinicie o computador. Reinicie o computador.”}

15 {“Os recursos do dispositivo estão em conflito com os recursos de outro dispositivo, provavelmente causado por reenumeração.”}

16 {“O *Windows* não pode identificar todos os recursos usados por este dispositivo. Atribuir recursos adicionais para o dispositivo.”}

17 {“O *Hardware* é um dispositivo multifuncional e o arquivo .inf para o dispositivo está a fornecer informações inválidas sobre como dividir os recursos do dispositivo para os dispositivos filho. Atualize o *Driver*.”}

18 {“Reinstale os *Drivers* para este dispositivo. Atualize o *Driver*. Desinstale e reinstale o *Driver*.”}

19 {“O *Windows* não pode iniciar este dispositivo de *Hardware* porque as suas informações de configuração (no registo) estão incompletas ou danificadas. Desinstale e reinstale o *Driver*. Reverta para a configuração do registo válida mais recente.”}

20 {“O *Windows* não pôde carregar um dos *Drivers* para esse dispositivo. O carregador VxD (Vxdldr) retornou um resultado desconhecido. Por exemplo, pode existir uma incompatibilidade de versões entre o *Driver* e o sistema operativo. Atualize o *Driver*.”}

21 {“O *Windows* está a remover este dispositivo. Atualize a exibição do Gestor de dispositivos. Reinicie o computador.”}

22 {“Este dispositivo está desativado. Ativar o dispositivo.”}

23 {“Falha do sistema. Se alterar o *Driver* do dispositivo não resolver, verificar a documentação do *Hardware*.”}

24 {“Este dispositivo não está presente, não está a funcionar corretamente ou não tem todos os controladores instalados. Atualize o *Driver*. Remova o dispositivo.”}

25 {“O *Windows* ainda está a instalar o dispositivo. Para completar a instalação, clique em Reiniciar para reiniciar o seu computador.”}

26 {“O *Windows* ainda está a instalar o dispositivo. Para completar a instalação, clique em Reiniciar para reiniciar o seu computador.”}

27 {“O *Windows* não conseguiu especificar os recursos para esse dispositivo. Clique na guia de Recursos e selecione a configuração básica para os recursos que esse dispositivo utiliza. Para ver quais os recursos que esse dispositivo utiliza, veja a documentação do dispositivo.”}

28 {“Os *Drivers* para este dispositivo não estão instalados. Instale o *Driver* de dispositivo.”}

29 {“Este dispositivo está desativado porque o *firmware* do dispositivo não forneceu os recursos necessários. Ativar o dispositivo na BIOS.”}

30 {“Este dispositivo está a usar um recurso de solicitação de interrupção (IRQ) que já está a ser usado por outro dispositivo e não pode ser compartilhado. Altere a configuração em conflito ou remova o *Driver* que está a causar o conflito.”}

31 {“Este dispositivo não está a funcionar corretamente porque o *Windows* não pode carregar os *Drivers* necessários para este dispositivo. Atualize o *Driver*.”}

32 {“Um *Driver* (serviço) para este dispositivo foi desativado. Desinstale e reinstale o *Driver*. Altere o tipo de inicialização no registo.”}

33 {“O *Windows* não pode determinar quais os recursos necessários para este dispositivo. Configure ou substitua o *Hardware*.”}

34 {“O *Windows* não pode determinar as configurações para este dispositivo. Configure manualmente o dispositivo.”}

35 {“O Firmware do sistema do computador não inclui informações suficientes para configurar apropriadamente e usar este dispositivo. Entre em contato com o fabricante para atualizar a BIOS.”}

36 {“Este dispositivo está a solicitar uma interrupção PCI, mas está configurado para uma interrupção ISA (ou vice-versa). Alterar as configurações para as reservas de IRQ.”}

37 {“O *Windows* não pode inicializar o *Driver* de dispositivo para este *Hardware*. Desinstale e reinstale o *Driver*.”}

38 {“O *Windows* não pode carregar o *Driver* de dispositivo para este *Hardware* porque uma instância anterior do *Driver* do dispositivo ainda está na memória. Execute o Assistente de solução de problemas. Reinicie o computador.”}

39 {“O *Windows* não pode carregar o *Driver* de dispositivo para este *Hardware*. Desinstale e reinstale o *Driver*.”}

40 {“O *Windows* não pode aceder a este dispositivo porque as suas informações de chave de serviço no registo estão ausentes ou gravadas incorretamente. Desinstale e reinstale o *Driver*.”}

41 {“O *Windows* carregou com êxito o *Driver* deste dispositivo mas não o consegue localizar. Atualize o *Driver*. Desinstale e reinstale o *Driver*.”}

42 {“O *Windows* não pode carregar o *Driver* para este *Hardware* porque existe um dispositivo duplicado a ser executado no sistema. Reinicie o computador.”}

```
        43 {“O Windows interrompeu o dispositivo pois este reportou problemas.
        Execute o Assistente de solução de problemas. Verifique a documentação
        do Hardware.”}

        44 {“Uma aplicação ou serviço desligou este dispositivo de Hardware.
        Reinicie o computador.”}

        45 {“Atualmente, este dispositivo de Hardware não está conectado ao
        computador. Reconecte o dispositivo ao computador.”}

        46 {“O Windows não pode aceder a este dispositivo de Hardware porque o
        sistema operativo está a ser desligado. Nenhuma resolução
        necessária.”}

        47 {“O Windows não pode usar este dispositivo de Hardware porque ele
        foi preparado para remoção segura, mas não foi removido do computador.
        Reconecte o dispositivo ao computador. Reinicie o computador.”}

        48 {“O Software para este dispositivo foi bloqueado porque é conhecido
        por ter problemas com o Windows. Atualize o Driver.”}

        49 {“O Windows não pode iniciar novos dispositivos de Hardware porque
        a seção do sistema é muito grande (excede o limite de tamanho do
        registo). Desinstale os dispositivos que não estiver a usar.”}

        50 {“O Windows não consegue aplicar todas as propriedades para este
        dispositivo, que podem incluir informação que descreve as capacidades
        e configurações. Reinstale o dispositivo. É recomendável contactar o
        fabricante para obter um novo Driver.”}

        51 {“Este dispositivo está a aguardar que outro dispositivo ou
        conjunto de dispositivos inicie. Não há resolução para este
        problema.”}

        52 {“O Windows não pode verificar a assinatura digital para os Drivers
        necessários para este dispositivo. Execute o Assistente de solução de
        problemas. Atualize o Driver.”}

    }

    $format = @{Expression = {$_.Name}; Label = "Device Name"}, @{Expression
= {$_.HardwareID} ; Label = "HardwareID"}, @{Expression =
{$_.ConfigManagerErrorCode} ; Label = "Error Code"} , @{Expression =
{$ErrorDesc} ; Label = "ErrorDescription"}

    $FaultyDevicesBASEInfo = $FaultyDevices | ConvertTo-HTML $format  | Out-
File FaultyDevicesBASEInfo.html; #-Property Name, DeviceID, HardwareID,
Status, ConfigManagerErrorCode, CompatibleID, @{Expression = {$ErrorDesc} ;
Label = "ErrorDescription"}

    $FaultyDevicesMOREInfo = $FaultyDevices | Format-List | Out-File
FaultyDevicesMOREInfo.log

    try

    {

        $Path2file = (Get-ChildItem $env:windir\inf -include oem*.inf -
recurse | select-String -pattern $FaultyDevice.Name | Select-Object -Unique
Path).Path.Split('.')[0]

        #Poderiamos desinstalar o .INF

        Remove-Item "$($Path2file +'.inf')" -Verbose

        #E o .PNF (tem de ser desta forma, porque no pnf não conseguimos
procurar o texto, visto que é um ficheiro pré-compilado)

        Remove-Item "$($Path2file +'.pnf')" –Verbose
```

```
        #Ou, usando o PNPUtil
           PnPUtil -f -d $Path2file
    }
    catch [System.Exception]
    {
        $_.Exception.Message | Out-File ExceptionMessage.log
    }
}
```

Anexo 2

Script para detetar *Drivers* Genéricos/*Standard*/*Microsoft*

```
#Obter informação sobre o modelo e SO
$manufacturer = Get-Content $Env:SystemRoot\logs\ComputerManufacturer.log
$Model = Get-Content $Env:SystemRoot\logs\ComputerModel.log
$OSArchitecture = Get-Content $Env:SystemRoot\logs\OSArchitecture.log
$OSName = Get-Content $Env:SystemRoot\logs\OSName.log

#Criar pasta para o modelo, apenas se ainda não existir (esta pasta servirá
tanto para os Drivers, como para os logs a analisar de cada modelo)
#Definir a localização corrente para esta pasta
Set-Location (Mkdir -p
"\\Driverserver\e\DriversRepository\$OSName\$OSArchitecture\$manufacturer
$Model" -Force)

try
{
    Out-File NonManufacturersDrivers.log
}

catch [System.Exception]
{
    $_.Exception.Message | Out-File ExceptionMessage.log
}

$words = 'Microsoft', 'Standard', 'Generic';

ForEach($word in $words)
{
    '### Dispositivos '+$word+' ###' | Out-File -Append
NonManufacturersDrivers.log
    if (!(Get-WmiObject Win32_PNPEntity | Where-Object{$_.Name -Match
$word}))
    {
        "`r`nNenhum dispositivo $word encontrado `r`n" | Out-File -Append
NonManufacturersDrivers.log
    }
    else
    {
        Get-WmiObject Win32_PNPEntity | Where-Object{$_.Name -Match $word} |
Select Name, HardwareID | Out-File -Append NonManufacturersDrivers.log
    }
}
```

Anexo 3

Script para verificar assinatura digital dos *Drivers*

```
$Drivers = Get-ChildItem '\\Driverserver\e\DriversFromManufacturers\' +
$maker + '\' + $Model -include oem*.inf, oem*.dll, oem*.exe –recurse

if($Drivers -eq $null)
{
    Write-Host 'Não existem ficheiros inf/exe/dll na pasta';
}
else
{
    Set-Location ("$env:SystemDrive\Program Files (x86)\Microsoft
SDKs\Windows\v7.1\Bin")
    ForEach($Driver in $Drivers)
    {
        try
        {
            .\Signtool verify /a /q $Driver | Out-File -Append
'\\DriverServer\e\DriversRepository\$OSName\$OSArchitecture\$manufacturer
$model\SigntoolVerification.log'
        }
        catch [System.Exception]
        {
            $_.Exception.Message | Out-File
'\\DriverServer\e\DriversRepository\$OSName\$OSArchitecture\$manufacturer
$model\ExceptionMessage.log' -force
        }
    }
}
```

Anexo 4

Script para instalar *Drivers pós-deployment*

```
$Repositorio =
“\\DriverServer\e\DriversRepository\$OSName\$OSArchitecture\$manufacturer
$model\DriversProcuraManual”

$Drivers = get-childitem -path $Repositorio -recurse -filter *.inf

foreach ($Driver in $Drivers)
{
   Write-host "A injectar Driver $Driver"
   pnputil -i -a $Driver.FullName
}
```

Abreviaturas

AD

Active Directory, 10

AIK

Windows Automated Instalation Kit, 30

BU

Business Unit, 5, 10

BYOD

Bring Your Own Device, 14

CMDB

Configuration Management Database, 9, 10

DHCP

Dyamic Host Configuration Protocol, 23

DISM

Deployment Image Servicing and Management, 24

DNS

Domain Name System, 11, 23, 30

DP

Distribution point, 29

DPM

Data Protection Manager, 27

DRP

Disaster recovery Plan, 9

GPO

Group Policy Objects, 15, 72

HVD

Hosted Virtual Desktop, 14, 15, 16, 19, 20

IAM

Identity and Access Management, 6

IIS

Internet Information Services, 29

ITIL

Information Technology Infrastructure Library, 27

MDT

Microsoft Deployment ToolKit, 16, 25

MOF

Managed Object Format, 13, 27

NAP

Network Access Protection, 30

NFG

Network For the Group, 6

NGWP

New Generation Workplace, 6, 7, 8, 9, 10, 12, 14, 16, 17, 18, 19, 20, 21, 22

PMP

Provisioning & Management Portal, 8

PS

Primary Site, 29

RAM

Random Access Memory, 15

RTM

Release To Manufacturing, 59

SCCM

System Center Configuration, Manager, 15, 16, 19, 22, 23, 26, 27, 28, 29, 30, 59, 60, 63, 65

SEP

Symantec Endpoint Protection, 19

SLA

Service Level Agreement, 8

SQL

Structured Query Language, 12, 13, 27, 29

SS

Secondary Site, 29

TI

Tecnologias de Informação, 6, 7, 9, 19, 20, 28

UCC

Unified Communications and Collaboration, 6

USMT

User State Migration Tool, 30

VMM

Virtual Machine Monitor, 27

VPN

Virtual Private Network, 15, 23

WDS

Windows Deployment Services, 30

WMI

Windows Management Instrumentation, 24, 30, 64

WSUS

Windows Server Update Service, 13, 19, 30

Bibliografia

[1] Managed Object Format (MOF),
URL:https://msdn.microsoft.com/en-us/library/aa8 23192%28v=vs.85%29.aspx

[2] VMware View,
URL:http://www.vmware.com/files/pdf/view/VMware-View-Dat asheet.pdf

[3] VMware ThinApp,
URL:http://www.vmware.com/br/products/thinapp/features.html

[4] Citrix XenApp,
URL:http://www.citrix.com.br/products/xenapp/overview.html

[5] Symantec Endpoit Protection,
URL:https://www.symantec.com/endpoint-protection/

[6] Group Policy objects,
URL:https://technet.microsoft.com/en-us/library/cc775691(v=ws.10).aspx

[7] AppLocker,
URL:https://technet.microsoft.com/en-us/library/ee424367(v=ws. 10).aspx

[8] BitLocker,
URL:https://technet.microsoft.com/pt-pt/library/cc766295(v=ws.10).aspx

[9] Conceitos de *Runbook*,
URL:http://technet.Microsoft.com/pt-br/Library/hh403820 .aspx

[10] ITIL,
URL: http://www.itil.org/en/vomkennen/itil/ueberblick/index.php

[11] BranchCache,
URL:http://technet.Microsoft.com/en-us/library/dd637832%28v= ws.10%29.aspx

[12] Descrição Geral do WMI,
URL: https://technet.microsoft.com/pt-pt/library/cc736575(v=ws.10).aspx

[13] Introduction to WDM , URL:https://msdn.microsoft.com/en-us/library/windows/
hardware/ff548158(v=vs.85).aspx

[14] Web-*Based* Enterprise Management (WBEM),
URL: http://www.dmtf.org/standards/wbem

[15] http://www.dmtf.org/standards/cim

[16] WMI Scripting Primer,
URL:https://msdn.microsoft.com/en-us/library/ms974579.aspx
[17] Microsoft - WMI Scripting Library,
URL: https://technet.microsoft.com/en-us/library/ee156554.aspx
[18] Arquitetura simplificada do Powershell,
URL: http://antapex.org/intro_powershell.htm
[19] How Windows PowerShell Works,
URL: https://msdn.microsoft.com/en-us/library/ms714658.aspx
[20] Truher, Jim. "Extend Windows PowerShell With Custom Commands". MSDN Magazine (Microsoft), Dezembro 2007.
[21] Introduction to Windows PowerShell Variables, URL:
http://www.computerperformance.co.uk/powershell/powershell_variables.htm
[22] Types of Cmdlet Parameters,
URL: https://technet.microsoft.com/en-us/library/dd878252(v=vs.85).aspx
[23] How PowerShell Formatting and Outputting REALLY works,
URL: http://blogs.msdn.com/b/powershell/archive/2006/04/30/586973.aspx
[24] More - How does PowerShell formatting really work?,
URL: http://blogs.msdn.com/b/powershell/archive/2006/06/21/641738.aspx
[25] Windows PowerShell Execution Policies,
URL: https://msdn.microsoft.com/en-us/library/ms714658.aspx#authorizationconcepts
[26] PowerShell.exe Console Help, URL: https://technet.microsoft.com/en-us/library/dd315276.aspx
[27] New-CMDriverPackage,
URL: https://technet.microsoft.com/en-us/library/jj821724(v=sc.10).aspx
[28] Get-CMDriverPackage,
URL: https://technet.microsoft.com/en-us/library/jj850190(v=sc.10).aspx
[29] Start-CMContentDistribution,
URL: https://technet.microsoft.com/en-us/library/jj850100(v=sc.10).aspx
[30] Get-CMCategory,
URL: https://technet.microsoft.com/en-us/library/dn151086(v=sc.10).aspx
[31] Get-CMCategory,
URL: https://technet.microsoft.com/en-us/library/dn151086(v=sc.10).aspx
[32] Import-CMDriver,
URL: https://technet.microsoft.com/en-us/library/dn151073(v=sc.10).aspx
[33] Command shell overview,

URL: https://technet.microsoft.com/en-us/library/bb490954.aspx
[34] Descrição geral do Windows *Script* Host, URL: https://technet.microsoft.com/pt-pt/library/4e236c2e-a484-4c1f-90bc-adb8d740dff4
[35] Wscript, URL: https://technet.microsoft.com/en-us/library/hh875526.aspx
[36] Cscript, URL: https://technet.microsoft.com/en-us/library/ff920171.aspx
[37] Command-Line Reference, URL: https://technet.microsoft.com/en-us/library/cc754340.aspx
[38] Informações do registo do Windows para utilizadores avançados, URL:https://support.microsoft.com/pt-pt/kb/256986
[39] Reg, URL:https://technet.microsoft.com/en-us/library/cc732643(WS.10).aspx
[40] Console Registry *To*ol for Windows (REG.EXE), URL:https://technet.microsoft.com/pt-br/library/cc668473.aspx
[41] Reg add, URL:https://technet.microsoft.com/pt-br/library/cc742162(v=ws.10).aspx
[42] Configure Automatic Updates using Registry Editor, URL:https://technet.microsoft.com/en-us/library/dd939844(v=ws.10).aspx
[43] Criar uma imagem personalizada do Windows PE,
URL: http://comunidadewindows.blogspot.pt/2012/10/criar-uma-imagem-personalizada-do.html
[44] Referência Técnica do Deployment Image Servicing and Management (DISM), URL: https://technet.microsoft.com/pt-pt/library/hh824821.aspx
[45] HP SSM, URL:http://h20331.www2.hp.com/Hpsub/downloads/SDM_SSM.pdf
[46] Lenovo, URL:https://support.lenovo.com/us/en/documents/ht037099
[47] ThinkVantage, System Update Solution Deployment Guide, Sixth Edition, Lenovo, 2014.
[48] Double Driver, URL:http://www.boozet.org/dd.htm
[49] Introduction to Code Signing, URL:https://msdn.microsoft.com/en-us/library/windows/desktop/aa380259(v=vs.85).aspx#introduction_to_code_signing
[50] Sign*To*ol, URL:https://msdn.microsoft.com/en-us/library/windows/desktop/aa387764(v=vs.85).aspx
[51] Microsoft Driver Verifier,

URL:https://msdn.microsoft.com/en-us/library/windows/hardware/ff545448(v=vs.85).aspx
[52] Verifier, URL: https://technet.microsoft.com/en-us/library/hh875581.aspx
[53] WinDbg Command-Line Options, URL: https://msdn.microsoft.com/en-us/library/windows/hardware/ff561306(v=vs.85).aspx
[54] Service Control Manager, URL: https://technet.microsoft.com/en-us/library/dd349449(v=ws.10).aspx
[55] LocalSystem Account, URL: https://msdn.microsoft.com/en-us/library/ms684190.aspx
[56] The LocalSystem Account, URL: https://msdn.microsoft.com/en-us/library/ms677973(v=vs.85).aspx
[57] *parâmetros* de linha de comando ("Command-line") para pacotes de actualização da Microsoft, URL: https://support2.microsoft.com/default.aspx?scid=kb;en-us;824687
[58] Como configurar o 'Gestor de dispositivos' para apresentar informações detalhadas, URL: https://support.microsoft.com/pt-pt/kb/304514
[59] Códigos de erro no Gerenciador de dispositivos no Windows, URL: https://support.microsoft.com/pt-pt/kb/310123
[60] Descrição de Códigos de Erro Gerados pelo Gerenciador de Dispositivos, URL: https://support.microsoft.com/pt-br/kb/125174
[61] Device Manager Error Messages, URL: https://msdn.microsoft.com/en-us/library/windows/hardware/ff541422(v=vs.85).aspx
[62] Opening a Dump File Using WinDbg, URL: https://msdn.microsoft.com/en-us/library/windows/hardware/hh451091(v=vs.85).aspx
[63] !analyze, URL: https://msdn.microsoft.com/en-us/library/windows/hardware/ff562112(v=vs.85).aspx